# A DESIGN SPACE EXPLORATION METHODOLOGY TO SUPPORT DECISIONS UNDER EVOLVING UNCERTAINTY IN REQUIREMENTS AND ITS APPLICATION TO ADVANCED VEHICLES

A Thesis
Presented to
The Academic Faculty

by

Christopher P. Frank

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy in the
School of Aerospace Engineering

Georgia Institute of Technology
August 2016

# A DESIGN SPACE EXPLORATION METHODOLOGY TO SUPPORT DECISIONS UNDER EVOLVING UNCERTAINTY IN REQUIREMENTS AND ITS APPLICATION TO ADVANCED VEHICLES

Approved by:


Prof. Dimitri N. Mavris, Advisor
School of Aerospace Engineering
*Georgia Institute of Technology*

Dr. Stéphanie I. Lizy-Destrez
Design and Control of Aerospace Systems Department
*ISAE-Supaéro*

Dr. Olivia J. Pinon-Fischer
School of Aerospace Engineering
*Georgia Institute of Technology*

Dr. Graeme J. Kennedy
School of Aerospace Engineering
*Georgia Institute of Technology*

Prof. Daniel P. Schrage
School of Aerospace Engineering
*Georgia Institute of Technology*



Date Approved: July 14, 2016

*To Dad and Mom*

# ACKNOWLEDGEMENTS

I am about to close this chapter of my life and I would like to take the time to thank and acknowledge the people who have helped pave the road and make this journey possible.

First, I would like to thank Prof. Dimitri Mavris, my academic advisor and mentor throughout the last few years. Doc, you have taught me far more in the past four years than I could have ever imagined and opened my eyes to the many different things I want to achieve in my career. Throughout my journey in the Aerospace Systems Design Laboratory, I have grown and learned skills that I know will tremendously help me in the future. Thank you for the trust you had in me and for offering me amazing opportunities, which collectively embody a unique learning experience. I also want to acknowledge Dr. Olivia Pinon-Fisher, my technical advisor and friend. Olivia, thank you for your advice, guidance, mentoring, and support throughout this journey. Thank you for spending days and nights reviewing and editing my dissertation. I would like to thank my other thesis committee members, Dr. Stéphanie Lizy-Destrez from ISAE-Supaéro, as well as Prof. Daniel Schrage and Dr. Graeme Kennedy from the School of Aerospace Engineering at Georgia Tech.

This research would not have been possible without the help of Clémence Tyl, Renaud Marlier, Frédéric Burgaud, Maxime Atanian, and Mathilde Deveraux. Clémence and Renaud, thank you for helping me implement the rocket engine model and the optimization algorithm, respectively. Fred, thank you for sharing your financial model with me. Max, thank you for your invaluable contribution to the Catia model. Mathilde, thank you for taking the time to read, comment, and review my dissertation.

I would like to acknowledge all the teachers I learned from since my childhood, I would not have been here without their guidance, blessing, and support. My special thanks to Clément Bastian and Eliane Fogelgesang. You have always supported me through all my endeavors and believed in me.

I also want to thank all my labmates in the Aerospace Systems Design Laboratory for

their moral support and motivation, which drive me to give my best. I really enjoyed working with everyone and having the opportunity to work in such a stimulating environment. I would especially like to thank Emmanuel Bourgin, Gokcin Cinar, Jean-Guillaume Durand, Clélia Level, William Levy, Yoshiki Miyairi, Jorge Calderon, Rosemonde Ausseil, Céline Bonicel, Yann Charront, Etienne Demers-Bouchard, Emmanuel Lacouture, Arturo Santa Ruiz, Damien Thioulouse, and Antoine Starck.

I would also like to extend my warmest thanks to my friends around the globe who witnessed and supported, through their visits, emails, messages or phone calls, the making of this work: Guillaume Erb, Aurélie Munch, Clotilde Grangé, Vincent Altmeyer, Laura Adam, Noelle Giersch, Laure Weiland, Noémie Funfrock, Ye Liu, Joachim Hodara, Nicolas Palpacuer, Héléne Evain, Mathieu Marant, Sébastien Courtiaud, Thomas Bour, and Laurent Muller.

Last but not least, I would like to express my deepest gratitude to my family. Special thanks go to Dad, Mom, Grannies Pierrette and Hermine, my godmother Marie-Christine, and my great-grandparents Opi and Omi who left us for a better world during this adventure. This dissertation would not have been possible without your warm love, continued patience, and endless support. Dad, Mom, I dedicate this thesis to you. I owe you everything and would not have made it this far without your unconditional love, patience, and unwavering support. Thank you, from the bottom of my heart, for encouraging me in all my endeavors. You have been, and will always be, my source of strength and inspiration, and I am proud to say that I live by your example.

# TABLE OF CONTENTS

# LIST OF TABLES

# LIST OF FIGURES

## List of Acronyms

**AAO** All-At-Once

**ABS** Antilock Braking System

**ACE** Astronaute Club Européen

**ACES** Advanced Cost Estimating System

**AEA** Association of European Airlines

**AHP** Analytical Hierarchy Process

**ALCCA** Aircraft Life-Cycle Cost Analysis

**ALTOS** Advanced Launcher Trajectory Optimization Software

**AMCM** Advanced Missions Cost Model

**ANN** Artificial Neural Network

**APEX** Association for Passenger EXperience

**ARM** Adaptive Reconfigurable Matrix of Alternatives

**ASDL** Aerospace Systems Design Laboratory

**ASTOS** AeroSpace Trajectory Optimization Software

**ATC** Analytic Target Cascading

**BIT** Bayesian Information Toolkit

**BMC** Bayesian Monte-Carlo

**BPR** Bypass Ratio

**CAD** Computer-Aided Design

**CAM** Computer-Aided Manufacturing

**CAPM** Capital Asset Pricing Method

**CBR** Case-Based Reasoning

**CDF** Cumulative Distribution Function

**CER** Cost Estimating Relationship

**CERs** Cost Estimating Relationships

**CFD** Computational Fluid Dynamics

**CO** Collaborative Optimization

**CPI** Consumer Price Index

**DAPCA** Development And Procurement Cost of Aircraft

**DARPA** Defense Advanced Research Projects Agency

**DOC** Direct Operating Cost

**DoE** Design of Experiments

**DSM** Design Structure Matrix

**EAI** Engine-related Attrition Index

**EBIT** Earnings Before Interests and Taxes

**EBITDA** Earnings Before Interests, Taxes, Depreciation, and Amortization

**ENVISAGE** Efficient Variable-Oriented Software for Architecture Generation

**ECLSS** Environmental Control and Life Support System

**EPA** Environmental Protection Agency

**ESA** Energy State Approximation

**FAA** Federal Aviation Administration

**FAI** Fédération Aéronautique Internationale

**FCF** Free Cash Flow

**FEM** Finite Element Method

**FPI** Fixed-Point Iteration

**FLOPS** FLight OPtimization System

**FOSM** First-Order Second Moment

**FTA** Fault Tree Analysis

**GA** General Aviation

**GMA** The General Morphological Analysis

**GTS** Generalized Trajectory Simulation

**HASA** Hypersonic Aerospace Sizing Analysis

**HRPS** Hybrid Rocket Propulsion System

**HTHL** Horizontal Take-Off and Horizontal Landing

**HTO** Horizontal Take-Off

**IAASS** International Association for the Advancement of Space Safety

**ICAO** International Civil Aviation Organization

**IDF** Individual Discipline Feasible

**ICR** Interest Coverage Ratio

**IOC** Indirect Operating Cost

**IRMA** Interactive Reconfigurable Matrix of Alternatives

**IRR** Internal Rate Of Return

**ISA** International Standard Atmosphere

**ISO** International Organization for Standardization

**JPDM** Joint Probabilistic Decision Making

**LEO** Low-Earth Orbit

**LRPS** Liquid Rocket Propulsion System

**LVCM** Launch Vehicle Cost Model

**MADM** Multi-Attribute Decision Making

**MCI** Mission Capability Index

**MDF** Multi-Disciplinary Feasible

**MDO** Multidisciplinary Design Optimization

**MEMIC** Morphological Evaluation Machine and Interactive Conceptualizer

**MER** Mass Estimating Relationship

**MFE** Model Fit Error

**MIT** Massachusetts Institute of Technology

**MOE** Measure Of Effectiveness

**MOGA** Multi-Objective Genetic Algorithm

**MRE** Model Representation Error

**NACA** National Advisory Committee for Aeronautics

**NASA** National Aeronautics and Space Administration

**NBI** Normal Boundary Intersection

**NHTSA** National Highway Traffic Safety Administration

**NNC** Normalized Normal Constraint

**NPGA** Niched Pareto Genetic Algorithm

**NPV** Net Present Value

**NSGA** Non-dominated Sorting Genetic Algorithm

**NSGA-II** Non-dominated Sorting Genetic Algorithm-II

**NWC** Net Working Capital

**OEC** Overall Evaluation Criterion

**OPR** Overall Pressure Ratio

**OSP** Orbital Suborbital Program

**OR** Operational Readiness

**OTIS** Optimal Trajectories by Implicit Simulation

**PAV** Personal Air Vehicle

**PCE** Product Cost Estimation

**PDF** Probability Density Function

**POST** Program to Optimize Simulated Trajectories

**PPES** Predator-Prey Evolution Strategy

**RASAC** Rapid Access-to-Space Analysis Code

**RBD** Reliability Block Diagrams

**RBCC** Rocket-Based Combined Cycle

**RCS** Reaction Control System

**RLV** Reusable Launch Vehicle

**ROCFLAM** Rocket Combustion Flow Analysis Module

**ROI** Return On Investment

**RPA** Rocket Propulsion Analysis

**RSC** Refurbishment/Spares Cost

**RSE** Response Surface Equation

**RSM** Response Surface Methodology

**SAIC** Science Applications International Corporation

**SEER** Systems Evaluation and Estimation of Resources

**SFC** Specific Fuel Consumption

**SOCS** Sparse Optimal Control Software

**SOSM** Second-Order Second Moment

**SP** Survivability Probability

**SPEA** Strength Pareto Evolutionary Algorithm

**SRLV** Suborbital Reusable Launch Vehicle

**SRM** Solid Rocket Motor

**SSSP** Space Shuttle Synthesis Program

**SSTO** Single-Stage-To-Orbit

**SWEEP** Structural WEight Estimation Program

**TGA** Thermodynamic Genetic Algorithm

**TIES** Technology Identification Evaluation Selection

**TIF** Technology Impact Forecasting

**TIT** Turbine Inlet Temperature

**TOC** Total Operating Cost

**TOGW** Take-Off Gross Weight

**TOPSIS** Technique for Order of Preference by Similarity to Ideal Solution

**TPS** Thermal Protection System

**TRF** Technology Reduction Factor

**TRL** Technology Readiness Level

**TSFC** Thrust Specific Fuel Consumption

**TSSP** Trajectory Synthesis Simulation Program

**TSTO** Two-Stage-To-Orbit

**UAV** Unmanned Aerial Vehicle

**USAF** United States Air Force

**USCM** Unmanned Space Vehicle Cost Model

**VBA** Visual Basic for Applications

**VEGA** Vector Evaluated Genetic Algorithm

**VTHL** Vertical Take-off and Horizontal Landing

**VTVL** Vertical Take-Off and Vertical Landing

**VTO** Vertical Take-Off

**VTOL** Vertical Take-Off and Landing

**WAATS** Weight Analysis of Advanced Transportation Systems

**WACC** Weighted Average Cost of Capital

**WATE** Weight Analysis of Turbine Engine

**WSA** Weighted-Sum-Approach

# SUMMARY

Recent technological developments have resulted in the emergence of new advanced vehicles such as suborbital vehicles and personal air vehicles. These innovative vehicles have in turn opened up new markets that are characterized by a large and complex solution space that requires designers to account for multiple objectives in early design phases. The complexity of these new vehicles also gives rise to a large combinatorial space of possible configurations for which no baseline has been established. A successful market penetration requires designers to define an optimized baseline and to identify both the main design drivers and potential technology gaps. Another major challenge is the presence of evolving uncertainty in requirements due to the lack of experience and established regulations. Hence, flexible decision-making techniques are needed to alleviate risks inherent to the launch of new programs and support informed go/no-go decisions. This research aims at supporting the development of emerging markets by establishing a methodology that enables a broad design space exploration at a conceptual level, and guides the selection of solutions against unclear objectives and under evolving uncertainty in requirements. In particular, this research uses the development of profitable, safe, and robust suborbital programs as a proof-of-concept to demonstrate the capabilities of the proposed methodology.

A review of current design approaches identified a lack of efficient design space exploration techniques. Current methods are indeed only capable of either comparing, at a high-level, numerous architectures or of optimizing a handful of alternatives with respect to more detailed parameters. In addition, there is a lack of available methodology to model and propagate evolving uncertainty in requirements. To bridge these gaps, a four-step methodology is developed based on the generic top-down design decision support process. First, the decision criteria are established. In particular, the design objectives are clearly identified and the design constraints are determined and modeled with time-dependent membership functions. Second, a new variable-oriented morphological analysis is developed to generate

all feasible concepts so that they can be systematically further optimized and compared. Third, a modeling and simulation environment is developed, which is capable of rapidly evaluating the performance, life-cycle costs, and safety of all types of suborbital vehicles at a conceptual design level. Finally, a new evolutionary multi-architecture algorithm based on architecture fitness is implemented that drives multi-objective optimization algorithms to simultaneously compare and optimize all configurations. To support decisions under evolving uncertainty, requirements are modeled with time-dependent membership functions and are propagated using fuzzy set theory.

The new modeling and simulation environment was developed and implemented in the context of suborbital vehicle design. By leveraging cycle-based approaches and surrogate modeling techniques, the performance of all chemical rocket engines can be evaluated with an accuracy of 3%, while dividing the execution time by a factor of $10^5$ compared to current physics-based models. This environment is also the first of its sort capable of estimating the life-cycle costs of hybrid rocket engines. The application of the proposed methodology also provides decision makers with key insights into the suborbital market. In particular, it demonstrates that a wisely developed commercial suborbital program might be profitable. The methodology also quantifies the trade-offs between affordable winged air launched vehicles powered by solid engines and safe slender vehicles powered by hybrid engines. When compared with existing approaches, the proposed methodology allows decision makers to find solutions 40% more performant for the same execution time or 40 times faster for the same accuracy. By quantifying the trade-offs between risk and expected performance, this methodology also helps designers make challenging go/no-go decisions and provides them with the best program start date. In particular, it provides a robust solution that increases the probability of success by 10% compared to those generated by traditional approaches. Finally, this methodology is a precious tool for designers who wish to quantify and rapidly assess the impacts of potential future regulations on the selection of a design concept and the profitability of the corresponding program.

# CHAPTER I

# MOTIVATION

The transportation of people and goods is crucial for a healthy economy and is one of the key contributors of overall good living standards. To meet people's needs, multiple means of transportation are available that are characterized by specific advantages and drawbacks. Requirements can be driven by various sources such as cost, duration, distance, topology, climate, and safety. To meet these requirements, more and more sophisticated vehicles have been developed over the last decades such as trains, cars, boats, aircraft, and spacecraft. Recent technology enhancements in multiple fields combined with demanding customers' requirements also result in the development of advanced vehicles which have the potential to open new markets: suborbital vehicles, flying cars, hypersonic commercial aircraft, magnetic trains, etc. To support the development of these emerging markets, decision makers must deal with specific challenges and characteristics. This chapter intends to identify these key drivers by analyzing emerging transportation markets with a particular emphasis on the ones served by suborbital vehicles and flying cars. A suborbital flight is a short flight (usually less than three hours) during which the vehicle reaches the Karman line. The latter is an imaginary boundary (that separates the atmosphere from outer space) set by the Fédération Aéronautique Internationale (FAI) at 100 kilometers (62 miles). In order to reach this limit, the vehicle must reach a speed high enough (around Mach 3) to perform a parabola before safely returning to Earth. The main difference with an orbital flight is that the maximum speed remains below the first cosmic speed (around 8 km/s), which is the speed required to escape Earth's gravity. Flying cars, also referred to as roadable aircraft intend to enhance door-to-door mobility while reducing their environmental footprint. These are hybrid vehicles which share characteristics with both aircraft and cars. On the one hand, they allow their owners to use conventional roads to reach areas which are not equipped with runways or which are too dense to enable take-offs and landings.

On the other hand, they also allow them to rapidly cross large forests, maritime areas or cover long distances (up to 900 km) faster than any conventional ground vehicles. To fully describe the specific features of these new markets, Section 1.1 highlights the importance of considering multiple objectives and their consequences on the vehicles' design. Next, Section 1.2 provides a review of existing concepts of suborbital vehicles and flying cars to illustrate the challenges inherent to the design of such vehicles. Then, Section 1.3 addresses the uncertainty in requirements and its evolution through the establishment of these new markets. Finally, Section 1.4 uses these challenges to establish decision makers' needs to support the development of those markets.

## *1.1 The Presence of Multiple and Competing Objectives*

When designing future aerospace vehicles, decision makers have to deal with multiple and competing objectives in addition to traditional performance objectives. Indeed, according to Schrage [386], 37% of project failures or delays are related to requirement definition problems: 13% come from poor user input elicitation, 12% from incomplete requirements, and 12% from changes in requirements. Therefore, requirements elicitation and analysis is a critical step of any design project or program. Yet it is a very challenging one that has serious consequences on programs' competitiveness and viability if not done properly [386]. To remain competitive, companies have to develop innovative products that successfully attract customers and meet both design and marketing requirements. As the product progresses through its life-cycle, the number of requirements and objectives to be met tends to increase, leading to more complex decisions. Examples of requirements and objectives in the context of the design of revolutionary concepts include affordability, safety, and customer satisfaction. These objectives, along with their impacts on the design, are presented in this section.

### 1.1.1 Affordability: a Key Enabler

The success of a market is primarily driven by its ability to meet potential customers' expectations, especially as they relate to cost. This concept is known as affordability, which is "the balance of benefits provided or gained from the system to the cost of achieving those

benefits" [281]. If only few customers can afford the product, its market penetration will be limited. The goal of this section is to demonstrate the importance of designing affordable products. For that purpose, the relationship between price and demand, known as the price elasticity on demand, is first discussed. Then, the impact of including affordability on the vehicle design is assessed, as well as the importance of decreasing the life-cycle costs.

#### *1.1.1.1 Price Elasticity*

The law of demand states that the higher the price of a good or product, the less customers will purchase. According to Anderson et al., this is the "most famous law in economics, and the one that economists are most sure of" [14]. This section aims at demonstrating that the sustainability of revolutionary concepts, especially suborbital vehicles and flying cars, highly depends on their affordability. Advanced vehicles are able to serve two different types of markets, transport and tourism, which are characterized by numerous alternatives already available to meet people's needs. Hence, the abundance of available substitutes makes the demand for these vehicles highly sensitive to price. This elasticity $E_d$ is measured using Equation 1, where $Q$ is the demand and $P$ the price [329].

$$E_d = \frac{\frac{dQ}{Q}}{\frac{dP}{P}} \tag{1}$$

Except for some very specific cases, this elasticity $E_d$ is always negative. For example, if a 3% increase in price implies a 6% decrease in demand, the price elasticity of demand will be -2. Based on this index, Anderson et al. decomposed the products/services into three categories [14]:

- Inelastic products: products for which the price elasticity is greater than -0.8. They mainly include basic food and legal services.
- Approximately unitary products: products for which the price elasticity is close to -1. They mainly include housing, private education and television receivers.
- Elastic products: products for which the price elasticity is smaller than -1. They mainly include restaurant meals, airline travels and luxury automobiles.

Anderson et al. also show that specific brands of vehicles and foreign leisure travels have a price elasticity close to -4 and are thus extremely sensitive to price. Price sensitivity can also vary with time. Hence, according to Smith et al., the price elasticity of air travels is expected to increase over the next decades, as shown in Figure 1 [400].

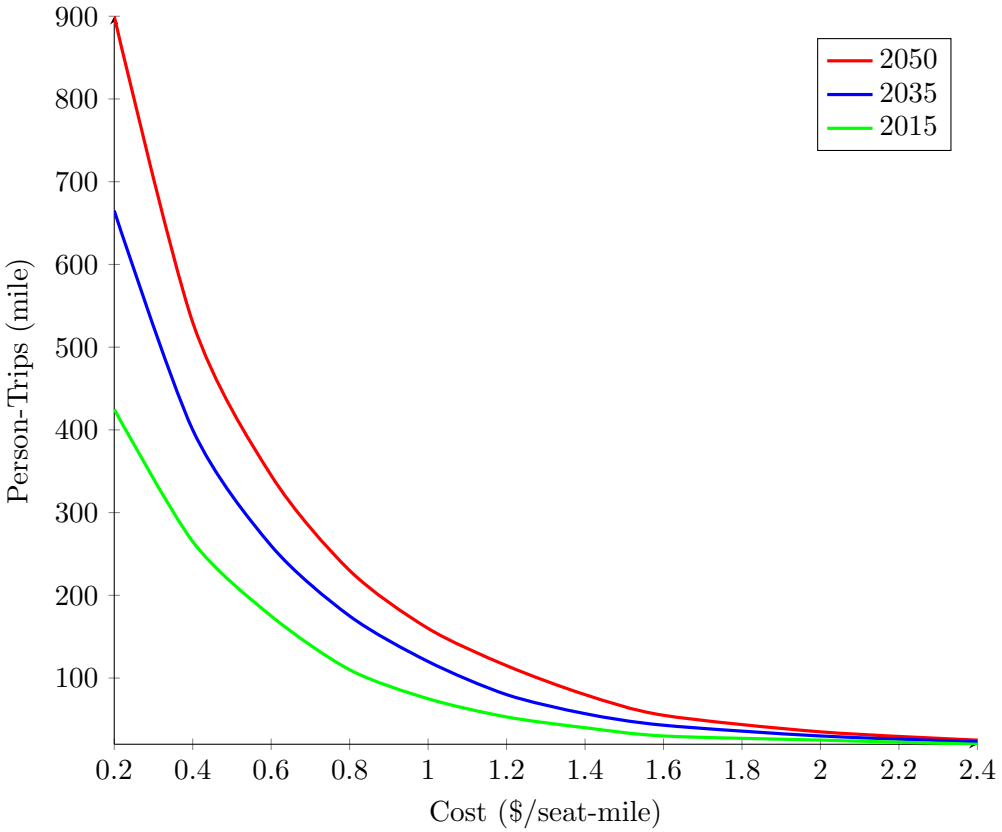



**Figure 1:** Price elasticity of air travels [400]

Mavris et al. emphasize the importance of cost reduction for emergent modes of transportation [255]. Figure 2 shows that if costs are not highly reduced, a new N+3 vehicle (vehicle that will reach a Technology Readiness Level (TRL) of 4-6 in 2025) will not be able to gain significant market shares. They also show that future high potential vehicles (with ticket price factor smaller than 3) have a price elasticity of -2.7.

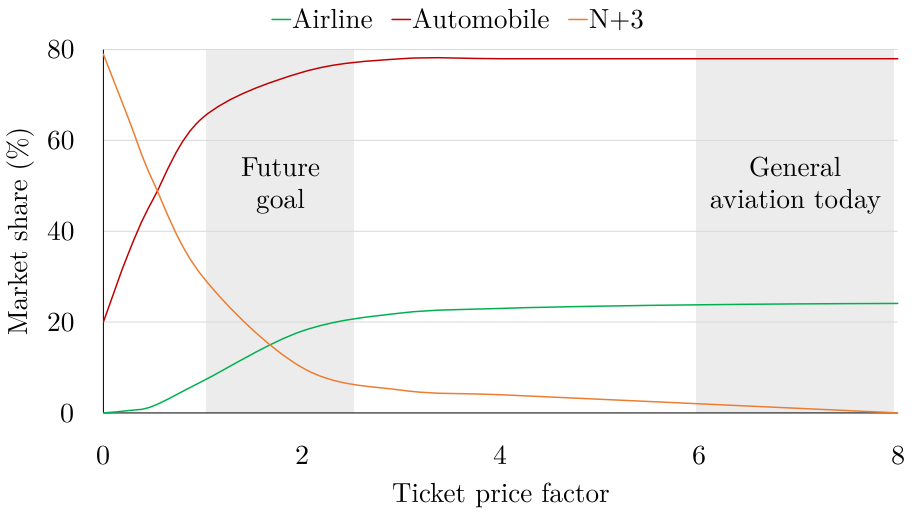



**Figure 2:** Importance of price reduction for market penetration [255]

Price elasticity for flying cars is even more important as such vehicles target a wealthy population wishing to improve its door-to-door mobility. Indeed, Ku shows that price elasticity is especially high when travels are driven by individual and luxury reasons rather than corporate and necessity reasons [246].

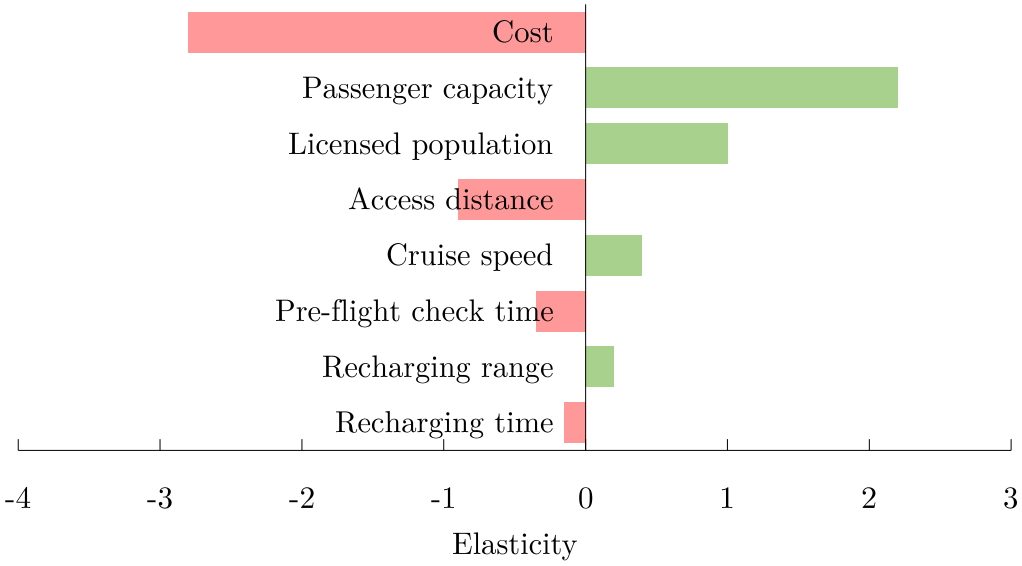



**Figure 3:** Key contributors to VTVL PAV market share [254]

Mavris et al. conducted a study to identify the key enablers of a Vertical Take-Off and Vertical Landing (VTVL) Personal Air Vehicle (PAV) market [254]. Presented in Figure 3, the results show that gaining market shares mainly relies on decreasing cost, which is the main driver. Passenger capacity, which can also be directly translated into cost, is the second key driver as it enables the cost to be distributed across more passengers.

Similar to flying cars, the successful development of the suborbital market also requires private companies and investors to keep the costs down to generate demand. The sensitivity of demand to ticket price for individuals with more than $5 million investable assets is illustrated in Figure 4.

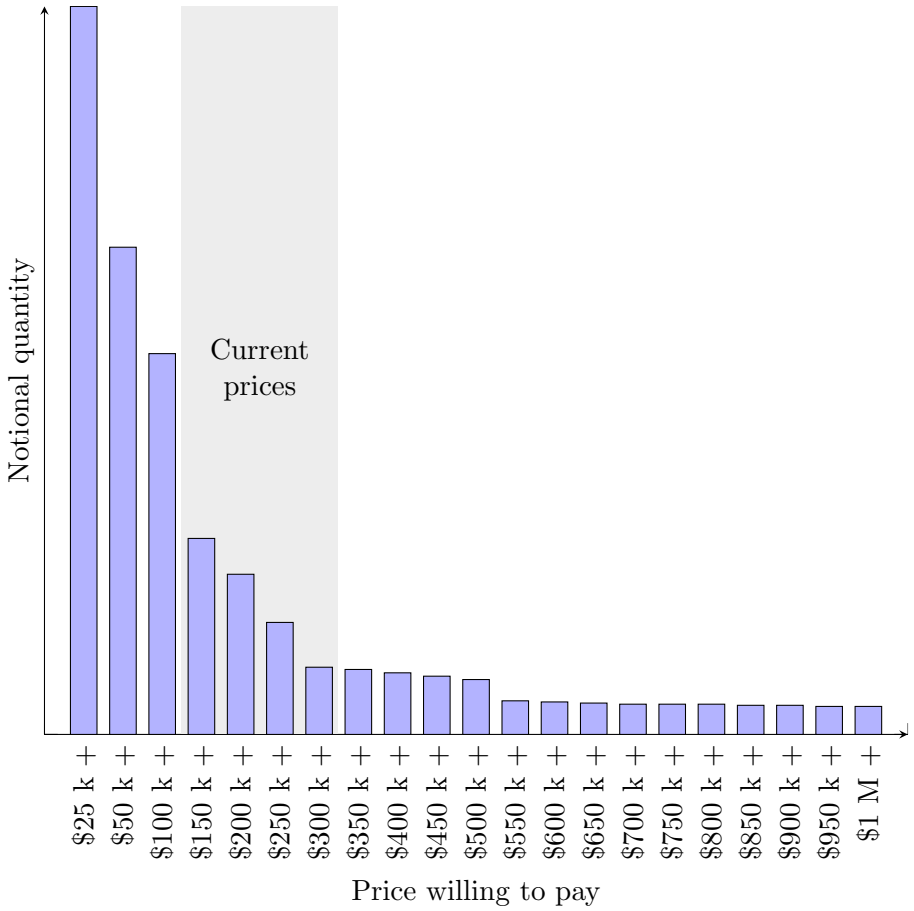



**Figure 4:** Price elasticity for suborbital tourism [435]

Virgin Galactic, XCOR, and Space Adventures anticipate a ticket price between $150,000 and $250,000 [459, 467, 473]. As illustrated in Figure 4, these prices are within the range where the demand is highly sensitive to small changes in prices. For ticket prices above $300,000, the market becomes a niche reserved to billionaires who are not really sensitive to ticket prices. Below this threshold, reducing the ticket price highly impacts the demand.

**Hence, the previous analysis shows that the markets served by new advanced vehicles are characterized by a demand highly sensitive to price.** Hence, in order to optimize profit, it is critical to understand the trade-offs between price and vehicle size.

#### *1.1.1.2 Impact of Demand on Design*

There are inherent correlations between demand, price and vehicle configuration. For example, lower ticket price results in higher demand but lower profit per ticket. Similarly, demand impacts the capacity and thus the size of the vehicle. Changes in vehicle size, in turn, impact the vehicle performance and consequently the ticket price. These relationships are represented in Figure 5.

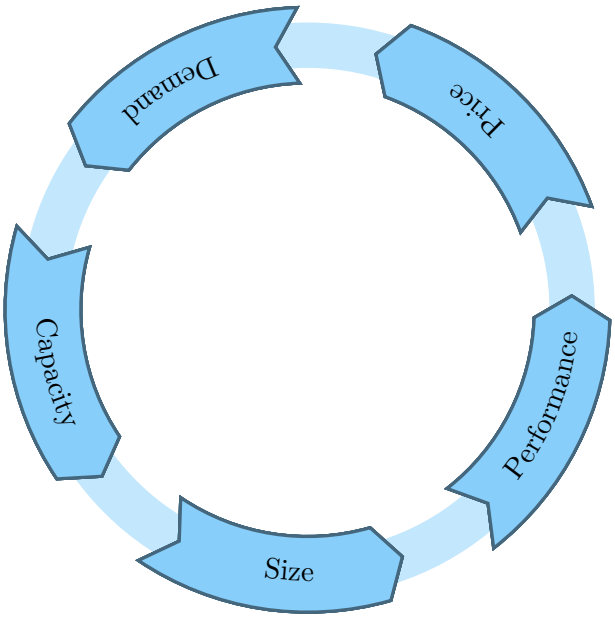



**Figure 5:** Relationships between demand, capacity, size, performance, and price

As demand increases and a new market is generated, regulations will have to be defined and put in place, which will also significantly impact the design of the various vehicles.

Moreover, if the demand is high enough to create a large worldwide market, regulations will necessarily emerge and also impact the design of the different vehicles. In particular, regulations that limit emissions, noise, etc. will be defined. Kroo et al. showed that the optimization of a wide-body aircraft against either cost or NOx reduction leads to very different vehicles [388, 453]. For example, the wing area varies from 3,890 $ft^2$ for minimum cost to 4,810 $ft^2$ for minimum NOx, which corresponds approximately to a 20% increase in wing area. Similarly, the thrust required at sea level jumps from 67,700 lbs for minimum cost to 97,400 lbs for minimum NOx, which corresponds to an increase of around 40% in thrust. Fan et al. investigated the impact of changes in cruise speed on airline operations and economics [139]. They showed that a 15% reduction in cruise speed would allow airlines to save up to 10% in fuel and consequently decrease their cost by approximately 8%. This decrease in cruise speed results in another optimum configuration. If new vehicles are spread around the world, specific local regulations such as noise limit are likely to emerge. Mavris et al. discussed the correlations between vehicle-level design variables and noise [253]. They highlighted that airframe design variables such as wing area and flap ratios have a high impact on approach noise whereas engine-sizing variables such as fan pressure ratio have a high impact on take-off noise.

In the context of emerging markets, such regulations have not been established yet. However, recent advancements require regulations and certification processes to be defined. Such stringent regulations will tend to slow the market expansion down [167, 265]. In addition, if new limits are established over the next decades to regulate new vehicles, they will impact the overall vehicle design and change the optimum design.

Depending on price and demand, a different public will be concerned: billionaire tourists, private pilots, general public, etc. Hence, the status of the passengers also impacts the design of the vehicle. Indeed, requirements are not the same for highly trained astronauts, fighter pilots, private pilots or general public. Therefore, the price level, and consequently the targeted market, will also influence the design of the vehicles. In 2004, the commercial

Space Launch Amendments Act introduced the first legislation for passenger transportation in space, but it is only applicable in the United States. Moreover, this legal framework is not complete yet and will continuously evolve with the introduction of new vehicles, new countries, and new companies [143, 252]. The establishment of an international regulation or a change in the current U.S. legislation would highly impact the design of the vehicles, especially in terms of maximum admissible load factor, rocket engine characteristics, etc. As far as flying cars are concerned, their democratization would require the designers to focus on safety. This is achieved through stall speed, since the lower the stall speed, the safer the vehicle. Decreasing the stall speed requires a decrease in maximum take-off gross weight, which in turn highly impacts cost. Moreover, the democratization of flying cars would probably require training skills for pilots. While they first intend to target private pilots, enhancements in new technologies such as avionics could help open this market. Less trained pilots or even the general public that followed a specific short training could be able to use these flying cars.

**All these observations demonstrate that demand, which is highly sensitive to ticket price, would highly impact the design of the vehicles.** The interrelationships discussed above are further complicated by the increase in competition often seen with the democratization and profitability of new markets. Moreover, the potential profitability nature of these new markets, usually tends to increase the number of competitors, as discussed in the following section.

#### *1.1.1.3 Highly Competitive Markets*

Even though military or government incentives have helped gain precious knowledge and technical capabilities about these revolutionary vehicles, emerging markets are mainly driven by private entities. In the context of suborbital vehicles, companies from more than eight countries have developed their own concepts including the United States of America, Romania, the United Kingdom, Canada, Argentina, Russia, Israel, and France. Figure 6 displays the location of suborbital commercial spaceports able to operate these vehicles. The spaceports that are ready for suborbital flight operations are shown in green, while the

spaceports under construction, or planned (the construction has not been started yet but money has already been invested) are shown in red. Nine countries are involved in such projects or already benefit from active spaceports: the United States, Sweden, France, the Netherlands, Malaysia, South Korea, Russia, and the United Arab Emirates.

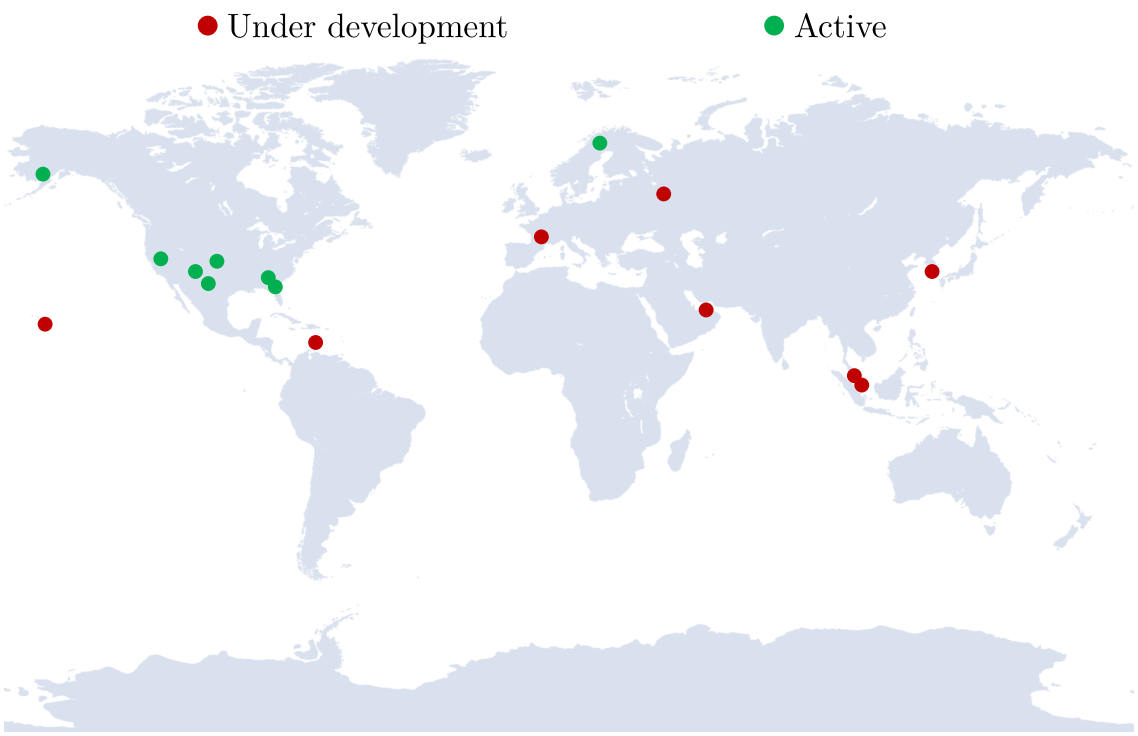



**Figure 6:** Location of current and future spaceports

Most of the spaceports are located in the United States since more than 50% of the current companies are based there. While being currently dominated by the United States, the market is expected to become a worldwide market in which a lot of money has already been invested and in which competition will be strong enough to see several competing companies emerge. Therefore, according to these studies, demand seems to support a worldwide emerging multibillion market in which numerous companies will design, build, and operate their own concepts over the next decades.

Similar to suborbital vehicles, hundreds of flying cars have been developed by entities such as National Aeronautics and Space Administration (NASA) and Defense Advanced

Research Projects Agency (DARPA). However, none has been certified or is commercially available yet. Among potential competitors, the three most advanced seem to be Terrafugia Transition [430], PAL-V [328], and AeroMobil 2.5 [6]. While the price is similar across all vehicles (around $300,000), they exhibit different design characteristics and performance.

This shift from government-funded projects to commercial enterprise is strongly characterized by a need for cost reduction. Indeed, companies and private investors do not have the luxury of cost and schedule overruns, overall life-cycle costs must be assessed and controlled and not passively suffered. Consequently, an important part of the design process will focus on reducing cost. For example, in 2011, SpaceX estimated the development costs of its Falcon 9v1.0 at approximately $390 million, while NASA had estimated these costs at $3.6 billion using its typical development process [121, 311, 411].

Hence, new markets are characterized by numerous competitors that develop their own concepts with specific capabilities based on various requirements. **As discussed, there is a need for reducing life-cycle costs while designing a vehicle that will face a strong competition.** If life-cycle costs are important when designing an aerospace vehicle, safety also has to be considered. Indeed, according to Jerome Lederer, first head of safety at NASA, "if you believe that safety is expensive, try an accident." [393] The need to include safety as a design objective is discussed in the next section.

### 1.1.2 Safety: a Regulatory Constraint

As most aerospace vehicles aim at transporting people and/or goods, safety becomes a critical requirement. Space programs such as Apollo, the Space Shuttle, and the Space Transportation Systems have demonstrated that iterative design methods based on performance analysis take time and can lead to concepts that would be too costly to produce or not safe to operate if they carry humans. During the 1990's, as reliability became a key requirement for customers, criteria used to select vehicles shifted from performance and cost to reliability. The Columbia Accident Investigation Board illustrates this shift: "The White House, Congress, and NASA leadership exerted constant pressure to reduce or at least freeze operating costs. As a result, there was little margin in the budget to deal

with the unexpected technical problems or make Shuttle improvements." [168] Since the Columbia disaster in 2003, NASA has promoted a safety-first policy that puts reliability and safety at the forefront of the design. As a consequence, this led to "architectures that meet vehicle and mission requirements for cost and performance, while ensuring that the risks to mission and crew are acceptable." [413]

Safety was a concern even before space programs. During World War II, E. Pieruschka explained that the weakest link theory was not sufficient to explain aircraft failures [261]. He showed that for a system with 20 components with a reliability of 90% each, the total system would have a reliability of only 12% [464]. Another example, from the building industry, is the CityCorp Center [476]. After it was built, a major flaw was discovered in the design: the building could not resist strong quartering winds if the tuned mass dumper was to shut down. Indeed, during the design, some adjustments were made, such as the use of bolts to secure joints instead of welding techniques. This led to a collapsing probability of 6.5%. Once the discrepancies were unveiled, CityCorp worked jointly with security services to develop an evacuation plan and make emergency repairs. Meanwhile, three weather specialists were monitoring winds. It shows that by making small changes in the design, the building's safety was not ensured anymore. The increased access to information has also impacted the design methodologies as failures are now quickly reported on the web. Public expectations are higher as capabilities are changing towards computerized resources that should allow faster and more reliable calculations. This demonstrates the importance of taking into account safety while designing in order to avoid extremely large operating and maintenance costs [452].

According to the Aerospace Corporation, the "launch of expendable vehicles, when used as a first stage to lift reusable rockets carrying crew and spaceflight participants, as well as launch and reentry of reusable launch vehicles with crew and spaceflight participants aboard, should be regulated differently than launch of expendable vehicles without humans aboard." [391] Hence, more stringent regulations are needed when vehicles carry human beings, especially paying passengers. Based on these observations, one can conclude that it is important to design for reliability in order to decrease the overall risk of failure [332].

In their forecast of the future market demand for U.S. suborbital reusable launch vehicles, NASA and the Aerospace Corporation identify safety as a key success factor. According to them, "societal demands for safety of transport vehicles have increased significantly over time. Because of this, assessing the level of safety that a system should achieve given its performance and technical maturity is a key to determining the demand for its services." [472] Besides, the Federal Aviation Administration (FAA)'s Office of Commercial Space Transportation recognizes the importance of integrating safety in the design process in order to promote the emergence of suborbital space tourism [264].

The International Association for the Advancement of Space Safety (IAASS) is working on providing rationalized guidelines using existing data, information, and knowledge originated from published papers, conferences, and industry resources [367]. These guidelines represent an effort to harmonize the different regulatory philosophies between the United States, where suborbital vehicles are licensed as launch vehicles, and Europe, where regulators have suggested to certify such vehicles as aircraft. At the May 7 meeting, though, suborbital vehicle manufacturers only expressed little support for the IAASS guidelines. "It's fundamentally wrongheaded," said Jeff Greason, chief executive officer of XCOR Aerospace, a company developing the Lynx suborbital vehicle. He argued it was unwise to establish safety standards through theoretical analyses rather than flight experience [155]. "I don't think it's right or healthy for the development of the industry to start touting these analyses as a level of safety we can achieve before we've achieved it." [155]

Most of the suborbital vehicle manufacturers are aware of the importance of safety and have adopted a design procedure that promotes reliability and safety. In particular, the Astronaute Club Européen (ACE) developed a vehicle following a "safety first" approach [190]. They considered safety and reliability as the chief goal of the project. For that purpose, they have selected a specific nontoxic kerosene and liquid oxygen engine that can be shut down in case of emergency. In addition, they have incorporated an air-breathing engine to improve the safety of the landing phase. Moreover, the cabin design is driven by safety requirements: reduction of effective load factor, ejection seats, flight control management system, etc. Similarly, the HyFlyer has been designed by trading life-cycle costs and operational

performance against system reliability, safety hazards, and environmental impacts [149].

Abensur developed innovative avionics and propulsion architectures for a "designed to safety" space vehicle [3]. For that purpose, he used segregation (physical separation between subsystems), differentiation (different hardware and software subsystem designs), and high failure tolerance.

In 2004, to support the development of small and easy-to-fly aircraft, the FAA created the light-sport category [146]. A light-sport vehicle needs to have a single engine, an unpressurized cabin and no more than two seats, weigh less than 600 kg, and have a maximum airspeed limit of 222 km/h. According to Allen, vice president of sales at Terrafugia, in 2008, engineers asked the FAA if they would consider a flying car as being under this regulation, "before they made the investment of time and effort they wanted to know that FAA would be supportive." [314] At that time, few flying cars existed and thus the FAA agreed. As a flying car can be operated both on roads and in the air, it needs to meet both regulations. When designing a roadable vehicle, it is necessary to take into account requirements for both the aircraft and the car parts. Those additional requirements may impact the design. For example, ground operations require airbags, windshields, wiping systems, impact guards, mirrors, etc. Those additional subsystems, in comparison with a simple aircraft, impact the empty weight. The structure of the flying car should also comply with crash test regulations. When in aircraft configuration, back-up systems such as full-vehicle parachutes should be implemented, also increasing the empty weight of the vehicle. Hence, summing the additional weight may exceed the limit imposed by the regulation. Designers will thus have to change the certification category, adding new requirements. For the Transition, they first started by meeting compliance requirements with the National Highway Traffic Safety Administration (NHTSA) by submitting simulation results [430]. One hurdle was establishing the responsibility between the requirements for air and road regulations. For example, its engine, designed for flying performance is not required to comply with federal emissions regulations, hence the Environmental Protection Agency (EPA) decided to consider it in the aircraft category. In 2015, Terrafugia was granted an exemption for its roadable aircraft by the FAA as its weight was over the regulation limit and by the NHTSA

as they wanted to change the windshield material from glass to lightweight plastic [258, 392]. Indeed, glass would not resist in case of bird strike while flying and cracks might reduce the operator's visual field. They also asked to change the tire type as they need to be more robust for take-off and landing, hence increasing the weight. Speed limitations also constrain the design: when on the road, the vehicle should be able to go at the same speed as other users, especially on the highway. When flying, maximum stall speed limitations should be ensured. Both those speeds constrain the design of the engine. In addition, as most roadable vehicles will be partially automated, subsystems will be necessary to avoid other flying elements in the air traffic, to fly in bad weather conditions, and to avoid restricted airspaces. Indeed, pilots will only have a twenty-hour training to be able to fly such a vehicle so they will not know all the limitations, that thus need to be computed by the vehicle. If the pilot becomes unresponsive, then emergency auto-landing should also be implemented [144, 317, 431].

All these observations and examples show that **safety is a critical design objective that needs to be considered in early design phases, as it highly impacts the design.** While being affordable and safe, a concept also needs to be marketable and meet consumers' expectations to be successful, as discussed in the next section.

### 1.1.3 Passenger Experience: a Key Competitive Advantage

Current economic conditions require designers to develop innovative products that perfectly meet customers' requirements. While multiple marketing approaches exist, one of the most commonly used for innovative products is the technology push strategy [208]. It is based on the design of new products due to enhancements in manufacturing techniques or technological capabilities. These changes result in an improvement of the products on the market, their cost, and performance. While this strategy tends to favor innovation, it often fails to adequately capture customers' needs. A good example of a technology push failure is the Sinclair C5 [34]: a small one-person electrically powered transport vehicle with pedal cycling assistance. This vehicle was developed in 1985 to be the first electric car. It was first launched in Great Britain and even though it was a state-of-the-art vehicle, it did not

receive great reviews and sales never went high. The public found it inappropriate for the British market as the C5 was not rainproof, could not go faster than 24 km/h [398], and needed pedaling for hill climbing. Hence, even though the chassis was designed by Lotus Cars and the material used was extremely innovative for 1985, the technology push strategy failed. To avoid such failures, another approach can be used: the market pull strategy. It is based on a clear and identified customer demand. This approach usually aims at generating a response to marketing actions. However, this approach fails to support the development of breakthrough products. As such, there is a need for an approach that considers both performance and the voice of the customer [159]. This section aims at discussing some of the additional requirements that have to be taken into account when designing future aerospace vehicles, especially the ones coming from customers.

According to many studies [16, 218, 334, 335], passenger preferences are based on their flight experience and comfort, which highly vary between airlines, seat categories, cabin layout, etc. Therefore, allowing airlines to improve passenger experience becomes a crucial competitive advantage for vehicle manufacturers. Even though cost is an important criterion, an improved comfort appears to have the same importance. According to Air Canada's CEO, "buying a fuel efficient aircraft is just half the recipe for pleasing passengers we need to differentiate." [241] In particular, the Association for Passenger EXperience (APEX) conducted a survey and discovered that, once on-board, the main concern for passengers is the legroom [480]. As the population is getting taller and bigger, seats are becoming too narrow and people are getting less comfortable, transforming an agreeable flight into a painful and stressful one [57, 227, 324, 470, 479, 483]. A survey was developed and published by Deveraux et al. [156] to identify passenger expectations about these aspects with a sample of 340 passengers. Seat comfort is an issue for 62% of passengers and 35% are willing to pay more for extra legroom.

As suborbital flights are usually short and expensive, passengers expect to benefit from a high level of comfort. Besides, since one of the key interests is to enjoy the micro-gravity, it becomes crucial for passengers to have enough space. Increasing the seat pitch, or legroom, will allow them to move freely and get a feeling of what astronauts live. When suborbital

flights are used for scientific research, the material needed might occupy a large volume in the cabin. The same objective has already been taken into account when designing aircraft for parabolic flights, as shown in Figure 7.


**Figure 7:** Experiments on board the A300 Zero-G [255]

Current manufacturers already advertise their larger seat pitch and cabin comfort compared to their competitors. For example, Virgin Galactic claims that: "SpaceShipTwo's cabin has been designed to maximize safety and comfort: it is the only spacecraft in history designed explicitly to optimize its passengers' experience. A dozen windows line the sides and ceiling of the spacecraft, offering each astronaut the ability to view the black skies of space as well as stunning views of the Earth below. Exposure to G-forces during SpaceShipTwo's ascent and descent is safely and comfortably managed thanks to systems such as our custom-designed, articulated seats, which are upright during rocket boost and reclined during reentry. The cabin is also designed for unfettered enjoyment of a large floating environment." [462]

As suborbital vehicles are mainly used for space tourism and scientific research, one of the key differentiation factors is their ability to provide a "space environment". This corresponds to the micro-gravity phase. Hence, one of the objectives when designing suborbital

vehicles might be to increase the duration of the weightlessness phase.

Another key metric when designing suborbital vehicles is the maximum acceptable load factor, which is directly linked to the force applied to the vehicle due to a change in velocity. Load factor directly impacts passengers and their comfort. A direct consequence of positive load factors is to make passengers feel heavier, and consequently to make motions difficult. In addition, load factors highly affect human organs and might cause passenger discomfort. This is caused by neurosensorial and cardiovascular effects, as well as other negative effects on lungs, human musculoskeletal system, inner ear, etc. As a consequence, limiting the maximum load factor usually becomes a key objective for manufacturers willing to improve passenger experience.

Table 1 compares these two metrics (seat pitch and duration of the micro-gravity phase) for different suborbital vehicle concepts. As demonstrated, these characteristics highly vary between manufacturers and might be used as competitive advantages.

**Table 1:** Passenger experience comparison for various suborbital vehicle concepts [252]

| **Metrics** | **Space-ShipTwo** | **RocketPlane XP** | **New Shepard** |
|---|---|---|---|
| **Seat pitch (m)** | 1.50 | 0.91 | 0.74 |
| **Weightlessness phase (min)** | 4 | 3-4 | 3 |
| **Maximum load factor (G)** | 3.5 | 5.0 | 6.0 |

For flying cars, the design is mostly centered on flying and driving easiness. Indeed, according to NASA's vision on personal air vehicles, they have to be "so easy to operate that, like a rental car, anyone with a driver's license can fly one." [94] This requires vehicle manufacturers to incorporate additional design objectives:

- Noise reduction: even though PAVs do not have their own noise regulations yet, it is expected that a maximum noise limit will emerge soon. This regulation is expected to follow the one applied for typical aircraft. The first aircraft noise regulation was adopted in 1969 with the introduction of noise certification standards (FAR Part 36).

International standards for aircraft noise were adopted in 1971 (ICAO Annex 16). The FAR Part 36 [145] sets aircraft noise standards in stages based on technology generation and production year, and establishes measurement procedures. Frank et al. [158] show that the maximum noise limit tends to decrease over time. Hence, reducing noise emissions might become an objective for future flying car manufacturers.

- Spacious cabin and bag trunk: similar to traditional aircraft and other means of transportation, the available volume is a key metric for passenger satisfaction. In addition, the space available for luggage also becomes important as flying cars are used for personal travels.

- Cabin accessibility and comfort: flying cars primarily target wealthy customers that tend to ask for a high level of comfort in the cabin. This corresponds to luxury and comfortable (often heavy) seats, cabin components, etc.

- Visual appeal: as PAVs are sold directly to private customers, visual appeal becomes a key differentiation factor between different vehicles. Even though aircraft and spacecraft traditionally rely on performance metrics, it is crucial to incorporate aesthetics in the design objectives of flying cars [142, 387, 446].

In addition, physical metrics such as load factors need to be taken into account. It should be minimized for take-off to maximize the passengers' comfort. According to Rogers [364], a load factor of around 0.5 g is acceptable as it reaches the same values in current cars.

Improving passenger experience tends to require a larger, heavier, and more complex vehicle [101, 300]. However, the competitive advantage gained is usually used to compensate the additional cost. According to Gulfstream, one of the core challenges for vehicle manufacturers is the "classic trade-off between delivering a leap ahead in terms of aircraft performance, balanced against the desire to combine this with superior cabin comfort." [458] For that purpose, Gulfstream included comfort as a core objective of their design through the assistance of their Advanced Technology Customer Advisory Team. This results in more than 200 design changes in both the cabin and the cockpit.

Hence, similar to life-cycle costs, **improving passenger experience is a key objective that also highly impacts the design of advanced vehicles.**

This section shows that, as systems become more and more complex and dynamic, additional requirements emerge. Furthermore, systems are now asked to become multi-functional, increasing the number of objectives to take into account [464]. Building on the previous observations, the following Assertion can be formulated:

**ASSERTION 1: Promising future markets are characterized by a multi-objective decision space, where trade-off analyses must be conducted in early design phases, as they might highly impact the vehicles' size and configuration.**

As discussed in this section, designing complex vehicles results in a complex decision space, where multiple objectives are usually competing. These objectives, arbitrarily prioritized by designers, lead to the emergence of a large number of possible vehicles. This rich diversity in concepts is another characteristic of such emerging markets, as discussed in the following section.

## *1.2 An Abundance of Designs*

New emerging vehicles tend to be extremely complex and are composed of numerous subsystems and functions. Contrary to well-established vehicles such as commercial aircraft or cars, their designs are still pushed by unconventional concepts and new combinations of technologies. Since no existing vehicles are currently in service, designers benefit from significant freedom and do not build their design around a baseline.

### 1.2.1 A Diversified Panel of Existing Vehicles

The development of new vehicles is often favored by the establishment of rewarded design competitions. Indeed, the first step of a full market penetration usually is the development of a working prototype. This enables companies to decrease risks and uncertainty related to potential markets while encouraging technology feasibility and enhancements. This was a

common thread in past aerospace applications and has promoted the development of many vehicles:

- Orteig Prize: first non-stop flight that links New York to Paris. This prize has been won by Charles Lindbergh with his aircraft the Spirit of St. Louis in 1927.
- Sikorsky Prize: first human-powered helicopter to meet a set of extremely challenging flight requirements. This prize has been awarded to Aerovelo's human-powered helicopter Atlas in 2013.
- Kremer Prizes: pioneers of human-powered flights.
- Lindbergh Electric Aircraft Prize: best electric aircraft.
- Ansari X Prize: first vehicle built by a non-governmental organization capable of carrying three people to 100 kilometers above the Earth's surface, twice within two weeks. This competition has been won by Scaled Composites with its SpaceShipTwo in 2004.

These competitions gathered a large number of competitors that developed extremely different concepts. To illustrate this variety, examples are given for suborbital vehicles and flying cars.

#### *1.2.1.1 Existing Suborbital Vehicles*

Created in May 1996, the Ansari X Prize was a true catalyst to the development of reusable suborbital vehicles. Funded by a large number of donors (Anousheh and Amir Ansari, First USA, etc.), this competition offered U.S. $10 million to the first non-governmental organization that "builds and launches a spacecraft capable of carrying three people to 100 kilometers above the Earth's surface, twice within two weeks" [484]. The competition was won on October 4, 2004 by the company Scaled Composites with its SpaceShipOne. The 26 competing teams developed concepts with various configurations tailored to different missions. These numerous concepts can be compared with respect to different criteria such as mission definition, propulsion system, and airframe configuration. Among the existing concepts, different types of launch techniques are used: typical

horizontal take-off from a runway, vertical launch, and air launch (by carrier aircraft or high-altitude balloons). As shown in Figure 8, there is no dominant launch type even if 44% of the concepts proceed by vertical launch.

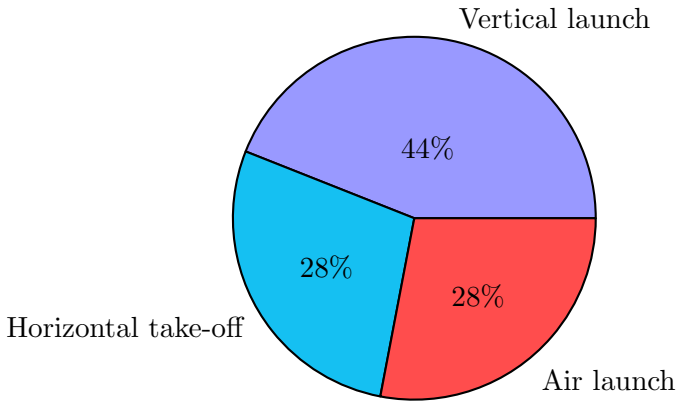



**Figure 8:** Different types of launch

Each technique has advantages and drawbacks that must be considered during the design process. A qualitative comparison of the various launch modes is presented by Marti and Nesrin Sarigul-Klijn [376, 377], F. Lehot and J-F Clervoy [252], the RAND Corporation [178], and N. Zakaria et al. [492].

Similar to the launch methods, there exists a wide variety of landing techniques but no dominant configuration. Figure 9 shows the proportion of the different methods. Gliding and horizontal powered landing are both horizontal landings on conventional runways (59% of the concepts), while rocket-powered landing and parachute braked landing are both vertical landings (41% of the concepts). In addition to these alternatives, Marti and Nesrin Sarigul-Klijn [377] also describe other types of aerodynamic decelerators such as parafoils and rigid or semi-rigid decelerators.

Each concept has a specific airframe configuration based on its launch and landing procedures. One can distinguish two major types: winged body or slender body. These airframe configurations often reflect the combination between the various types of launch and landing. For example, a vehicle that lands horizontally must have wings, even if it is

launched vertically. Some observations about these combinations can be made. When vertical launch is combined with horizontal landing, the landing gear only needs to be sized for the landing weight (usually significantly lower than the take-off gross weight). If a vehicle takes off and lands vertically, the load path is always axial and therefore the overall empty weight is reduced.

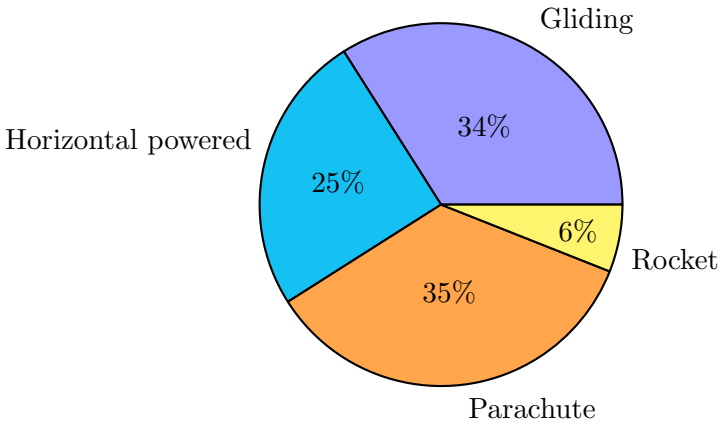



**Figure 9:** Different types of landing

Up to three different engines can be implemented on a single vehicle, each with a specific function based on the type of mission to be flown. Air-breathing engines are sometimes used for phases at low altitudes (high-density). Rocket engines are used for high thrusts and high altitudes during a short amount of time. Finally, attitude control systems must be used since aerodynamic-related devices such as ailerons are not efficient enough at high altitude.

All concepts are equipped with rocket engines: 21% are hybrid such as the SpaceShipOne, 13% are solid such as the Cosmopolis XXI, and 66% are liquid such as the Vehra. Moreover, among existing concepts, approximately 30% are equipped with both a rocket engine and an air-breathing engine. Even if this additional engine adds weight and complexity to the design, it is much more efficient during the first part of the climb, where the atmosphere is dense enough.

To illustrate this variability, Figure 10 shows two examples of the X Prize vertical take-off concepts: the Black Armadillo from Armadillo Aerospace and the Canadian Arrow from

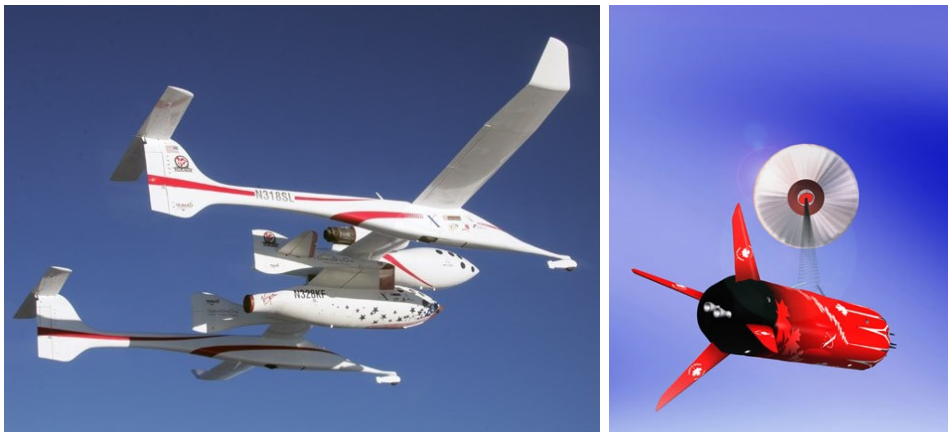
Canadian Arrow. Figure 11 presents two horizontal take-off concepts: the Rocketplane XP from Pioneer Rocketplane and the SpaceJet from Astrium. Finally, Figure 12 shows the SpaceShipOne from Scaled Composites and the Wild Fire from the da Vinci Project, which are respectively an aircraft-carried and a balloon-carried vehicle.

(a) Black Armadillo [130] (b) Canadian Arrow [24]

**Figure 10:** Examples of vertical take-off concepts

(a) Rocketplane XP [108] (b) SpaceJet [30]

**Figure 11:** Examples of horizontal take-off concepts



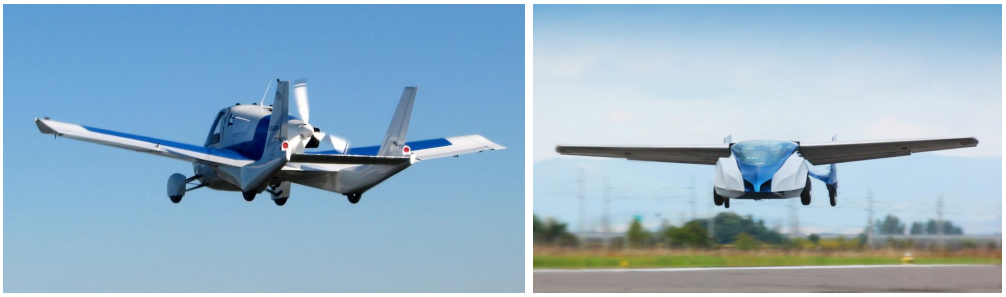
(a) SpaceShipOne [380] (b) Wild Fire [409]

**Figure 12:** Examples of air-launched concepts

#### *1.2.1.2 Existing Flying Cars*

The concept of flying car is not new and first emerged in 1917 with Glenn Curtiss. While the "Curstiss Autoplane" was able to take off, it never performed a full flight [456]. However, it marked the beginning of a long series of trials that can be illustrated by examples such as the Aerocar from Taylor in 1949 [428], the Airphibian from Fulton in 1950 [401], and Aerobile from Waterman in 1957 [456]. Nowadays, several companies are developing commercial prototypes. On the military side, DARPA is promoting the development of "innovative solutions" for military use: a roadable Vertical Take-Off and Landing (VTOL) vehicle capable of carrying up to four persons and their gear. Similar to the Ansari X Prize, this competition will reward winners with $9 million to support Phase 1 development. Various concepts have emerged with extremely different characteristics. Similar to suborbital vehicles, these numerous concepts can be compared with respect to different criteria such as mission definition, propulsion system, and airframe configuration. Among the existing concepts, both horizontal and vertical take-off/landing concepts have been designed. In order to enable vertical take-off, concepts must be equipped with either rotors or rotating engines, while horizontal take-off concepts rely on typical jet engines or propellers. This variability is characterized by trade-offs between efficiency in cruise and potential use of small areas to take off and land. The propulsion configuration may also vary between competitors:

tractor (forward-mounted engines), pusher (backward-mounted engines) or tractor-pusher (mixture). For winged bodies, the location of the wings with respect to the fuselage is another variable: low wing, mid-wing, high wing or even parasol wing. One of the key technological challenges is the ability to retract the wings. As such, four main concepts exist: detachable wing, variable swept wing, folding wing, and telescopic wing. Finally, the landing gear/wheel configuration also varies among existing concepts: tricycle or quad-cycle, which could be either retractable or fixed. These alternatives have been widely implemented and combined on existing vehicles so that no baseline seems to emerge. To illustrate this strong variety, Figures 13 and 14 describe some examples of existing concepts. Figure 13 shows two examples of horizontal take-off concepts: the Terrafugia Transition from Terrafugia and the Aeromobil 2.5 from Aeromobil. While they both have retractable wings, the wings of the latter rotate around the vertical axis and the wings of the former around the horizontal axis. Figure 14 presents two vertical take-off concepts: the PAL-V One from PAL-V and the Terrafugia TF-X from Terrafugia. The latter uses rotating propellers to enable a vertical take-off and an efficient cruise, while the former uses a rotor so that the vehicle behaves like an helicopter.



(a) Terrafugia Transition [430] (b) AeroMobil [6]

**Figure 13:** Examples of horizontal take-off concepts

This section showed the variety of existing configurations for each vehicle because of the large number of features described by numerous possible options. Hence, the various possible combinations result in a large combinatorial space that must be explored by designers, as discussed below.

(a) PAL-V One [328] (b) Terrafugia TF-X [430]

**Figure 14:** Examples of vertical take-off concepts

### 1.2.2 Large Combinatorial Space

The complexity of aerospace systems requires designers to select the best option for each subsystem. The large number of available options for each choice contributes to enlarge the design space. In addition, new markets require designers to develop both a configuration and its corresponding operations. Indeed, there is currently no clear request for proposal or design mission for such vehicles. The National Business Aviation Association provides a template for requests for proposal for aircraft charters [312]. It includes precise information about the mission, the number of passengers, the price, etc. However, no such document exists for emerging markets. Hence, this leads to enlarging the design space and prevents designers from locking on a design. As a consequence, designers need to carry more design alternatives longer throughout the design process. To illustrate this observation, a high-level morphological matrix is created for both flying cars and suborbital vehicles. These morphological matrices decompose both vehicles and missions into high-level features (rows) in order to identify each available option (columns). These matrices are presented in Tables 2 and 3. As shown in Tables 2 and 3, the number of alternatives that can be generated by combining the different options is extremely large. In addition, this first analysis does not include any detailed information. As a consequence, these matrices could be expanded even more and the corresponding number of alternatives would grow exponentially. While a compatibility analysis along with some qualitative considerations can be made to help decrease the combinatorial space at the beginning, it will still remain extremely large. **This large**

**number of possible configurations cannot be individually evaluated by designers. Hence, a rigorous and systematic methodology that enables quantitative trade-off analyses to support the selection of a design is needed.** In order to tackle this large design space, designers have relied on current practices, as discussed below.

**Table 2:** Morphological matrix for suborbital vehicles

| | **Alt. 1** | **Alt. 2** | **Alt. 3** | **Alt. 4** | **Number of alt.** |
|---|---|---|---|---|---|
| **Take-off method** | Horizontal | Vertical | Aircraft launched | Balloon launched | 4 |
| **Landing method** | Horizontal powered | Gliding | Rocket | Parachute | 4 |
| **Wing** | Delta | Swept wing | Straight wing | None | 4 |
| **Vertical surface** | Vertical stabilizer | Wing tip | None | | 3 |
| **Horizontal surface** | Horizontal stabilizer | Canards | None | | 3 |
| **Main engine** | Liquid | Solid | Hybrid | | 3 |
| **Number of rocket engines** | 1 | 2 | 3 | 4 | 4 |
| **Auxiliary engine** | Typical turbojet | Augmented turbojet | Typical turbofan | Augmented turbofan | 4 |
| **Number of auxiliary engines** | 0 | 1 | 2 | 3 | 4 |
| **Afterburners** | Yes | No | | | 2 |
| **Attitude control** | Cold gas | Liquid | | | 2 |
| **Number of possible combinations** | | | | | **442,368** |

### 1.2.3 Current Practices and Limitations

Due to the novelty of such vehicles, companies logically favored the use of technologies that they have already developed and leveraged their own internal skills and know-how to design their vehicle. During this design exercise, no design space exploration or global

**Table 3:** Morphological matrix for flying cars

| | **Alt. 1** | **Alt. 2** | **Alt. 3** | **Alt. 4** | **Number of alt.** |
|---|---|---|---|---|---|
| **Take-off method** | Horizontal | Vertical | | | 2 |
| **Landing method** | Horizontal | Vertical | | | 2 |
| **Wing** | Delta | Swept wing | Straight wing | None | 4 |
| **Vertical surface** | Vertical stabilizer | Wing tip | None | | 3 |
| **Horizontal surface** | Horizontal stabilizer | Canards | None | | 3 |
| **Aircraft engine** | Jet | Propeller | Rotor | Turboprop | 4 |
| **Number of aircraft engines** | 1 | 2 | 3 | 4 | 4 |
| **Car engine** | Diesel | Gasoline | Electric | Hybrid | 4 |
| **Rotating engines** | No | Horizontal axis | Vertical axis | | 3 |
| **Wing location** | Low | Mid | High | | 3 |
| **Propulsion type** | Tractor | Pusher | Tractor - Pusher | | 3 |
| **Landing gear** | Tricycle | Quad-cycle | | | 2 |
| **Number of possible combinations** | | | | | **497,664** |

optimization was performed. Instead, major design decisions were made based on expertise and company's experience. This lack of rigorous methodology has also been the "norm" in the design of launchers that emerged in the last decades [455]. Hence, each company tends to push for the configuration they believe in. Suborbital vehicles can be used to illustrate this statement. For instance, Bristol Spaceplanes and Armadillo Aerospace push for liquid engines, while Vanguard Spacecraft and Canadian Arrow push for solid engines, and Scaled Composites and Pablo de Leon & Associates for hybrid engines. Some companies such as Pioneer Rocketplane and Dassault Aviation believe in lifted bodies, while others (Armadillo Aerospace and Acceleration Engineering) favor non-lifted bodies. Bristol Spaceplanes led by

David Ashford (an aircraft aerodynamicist who also worked on rocket motors), developed a conventional aircraft augmented by a liquid rocket engine. Burt Rutan, who designed the SpaceShipOne, was an aircraft designer as well and developed a winged body. Richard Speck, who led Micro-Space, was a charter member in the National Association of Rocketry and designed a VTOL concept [484].

Roadable aircraft are another example of this diversity. Two design approaches are usually considered: enable cars to fly or enable aircraft/rotorcraft to roll. For example, Stefan Klein, chief designer and co-founder of AeroMobil built his vehicle around a car, after having spent years leading innovative research projects for automotive companies including Audi, Volkswagen, and BMW [6]. Designers from PAL-V decided to build their concept around a rotorcraft and were directed by Peter Jorna who worked 18 years at the National Aerospace Laboratory in Amsterdam and served as head of the Flight Division, managing Helicopters, Operational Research, Human Factors, Flight Mechanics, and Flight Simulation departments [328]. Finally, Carl Dietrich, who received his BS, MS, and PhD from the Department of Aeronautics and Astronautics at the Massachusetts Institute of Technology (MIT), founded Terrafugia and built his concept around an aircraft [430]. Once a baseline architecture has been selected, an optimization is only performed locally so that no strong similarities can be noticed between concepts. This lack of baseline is very specific to new emerging markets. For conventional commercial aircraft, a baseline is defined for each type of mission and many similarities exist between concepts. Hence, for emerging markets such as suborbital vehicles and roadable aircraft, the use of current practices based on local optimization leads to a large variety of configurations.

**Based on the previous discussion, one can conclude that relying on companies' expertise and qualitative considerations results in the development of various concepts among which no baseline has been defined.** These observations lead to the following Assertion:

**ASSERTION 2: A rigorous and systematic methodology is needed that enables the exploration of a large combinatorial design space and supports quantitative trade-off analyses to facilitate the selection of a design.**

The diversity in existing concepts, as discussed in this last section, can also be explained by the lack of regulation and clearly defined requirements. This aspect is further developed in the section below.

## *1.3 Evolving Uncertainty in Requirements*

The necessary regulatory frameworks for these growing markets are currently being discussed but have not been fully defined yet. Indeed, because the technologies involved in new activities are rapidly expanding, there is a strong need to develop an appropriate regulatory framework. Finally, it is important to note that, due to the novelty of the concepts, the possible applications have not yet been completely defined and the offered vehicles are designed for experiences that highly vary among the potential competitors.

### 1.3.1 A Confusing Regulatory Framework for New Vehicles

Emerging markets are usually characterized by the use of new technologies as well as hybrid solutions that could highlight gaps in current regulations. Revolutionary technologies such as those used for magnetic levitation of trains require precise regulations in terms of certification and safety. Suborbital tourism may be one of the trickiest markets to regulate. Indeed, while multiple new technologies have to be certified, the legal framework is also hybrid as it is at the boundary between space laws and air laws. Another hybrid market is the roadable aircraft. In this case, the vehicle is at the boundary between two well-regulated domains: ground transportation and air transportation. These two markets will be investigated in detail to further illustrate the lack of well-defined requirements for emerging markets.

#### *1.3.1.1 Suborbital Vehicles*

Suborbital flights will be performed by either an aircraft or a spacecraft. Independently of the concept used, it needs to reach an altitude which corresponds to the limit between the atmosphere and outer space. Therefore, an important question is naturally raised: should the space law, the air law or even both laws be applied? The answer to this question will have significant legal implications on suborbital space tourism. Beginning in 1984 with the

Commercial Space Launch Act, the legislation concerning commercial space transportation mainly focused on public health and safety issues. Then, in 1998, the FAA was granted authority for licensing the returning vehicles from space. Two years after, the FAA's Office of Commercial Space Transportation finalized its licensing process for reusable launch vehicle missions including crew and payload. On the civil aviation side, the legislation is regulated by both the FAA and the Aviation Safety. These two legislatures (aircraft and spacecraft) have always evolved independently from each other. Therefore, the regulation hybrid vehicles (aircraft airframe and rocket engines, air launched vehicles, etc.) fall under is not clear cut. Indeed, as mentioned by Axelle Cartier and Ioana Cristoiu [71], "suborbital flights are at present the major issue to be dealt with under the current legislation. [...] It is not clear what will be the applicable law. It is to be expected that issues will depend on different national legislatures." Even if the FAI considers the Karman line at 100 kilometers as being the limit with outer space, there is no clear physical line. The definition of the layer contained between 80 km and 110 km is still debatable [180, 463, 465, 482].

In 2004, the commercial Space Launch Amendments Act introduced the first legislation for passenger transportation in space but it is only applicable in the United States. Moreover, this legal framework is not complete yet and will continuously evolve with the introduction of new vehicles, new countries, and new companies [143, 181, 252]. Also, the establishment of an international regulation or a modification of the current U.S. regulation is very likely to impact the design of the vehicles, especially in terms of maximum admissible load factor, rocket engine characteristics, etc.

In addition, while the flight is the most important part of the suborbital experience, ground infrastructures are needed and will contribute to the passenger training, the maintenance of the vehicles, the storage of components, launch and recovery platforms, etc. A dilemma appears between the use of existing airports and the development of new and better-suited spaceports. If the vehicle is registered as an aircraft, it can be operated from existing airports. Nevertheless, several issues still remain:

- Vehicles must behave like an aircraft and must be able to follow the current air traffic rules.

- Maintenance must be done on rocket engines and any unconventional components using existing machines and infrastructures.
- Environmental requirements such as noise and pollution must be met.
- Airport safety services and procedures must be efficient enough to ensure the safety of suborbital operations.

Most of these challenges can be avoided by building customized, but expensive, spaceports. Moreover, if the vehicle takes off horizontally and uses a runway, the latter must be certified by an airport regulation. If the vehicle is launched vertically, no regulation exists for commercial launch pads. As such, the establishment of regulations concerning ground infrastructures will also have an impact on the design and the mission of suborbital vehicles.

The legal regime applicable on board the vehicle depends on its registration status: aircraft or space object. This becomes even more important for space tourism activities. Indeed, these new passengers will face many medical constraints that would probably require the establishment of a compulsory pre-flight medical examination whose guidelines have still to be determined. Moreover, in order to improve the passengers' experience and safety, the following constraints have to be addressed during the design phase:

- High g-forces: during the rocket powered acceleration phase and the reentry phase, the vehicle undergoes a significant load factor which could create gray-out (partial loss of vision), blackout (loss of vision), G-LOC (G-force induced Loss Of Consciousness) or even death. The limits highly depend on the passenger and his/her level of fitness. According to NASA [99], untrained humans can support up to 20 g during less than 10 seconds or 6 g during 10 minutes. Blackout often occurs between 4 and 6 g. These high g-forces can also increase the heart rate, causing a pain in the human musculoskeletal system and dyspnea. The alternation of high and low load factors tends to favor cardiovascular diseases such as aneurysm and could lead to disastrous ends especially for people already subject to slight heart diseases. Hence, for health and comfort-related issues, this load factor has to be highly reduced. Some design considerations would help reducing these effects. For example, reclining seat and

low acceleration rate would increase the comfort of the passengers. Anti-g suits are another option to reduce the effects of high load factor. While increasing the payload weight and reducing the passenger's comfort, these suits could improve the resistance of passengers and consequently allow them to withstand higher load factors.

- Exposure to radiations: the vehicle travels in the upper-level of the atmosphere where galactic and solar radiations are more intense. According to the Health Physics Society [252], radiations received during a suborbital flight are 0.0053 mSv. This value is five times lower than a round trip transatlantic flight and twenty times lower than a lung radiography.

- Vibrations: during take-off and reentry, the vibrations undergone by passengers could become annoying for sensitive passengers and must therefore be reduced.

**All these constraints are not being regulated yet and must be considered as uncertainty sources in the design requirements. However, with the progressive establishment of the regulatory framework, this uncertainty will tend to decrease and constraints to converge towards fixed values.**

#### *1.3.1.2 Flying Cars*

Since the beginning of the 21$^{st}$ century, technologies seem mature enough to enable the design of viable flying cars. This is confirmed by the numerous prototypes designed around the world. While viable vehicles have already shown their potential, government officials and designers must decide which regulations to use: aircraft, automotive, both or even a completely customized one. Potential regulatory frameworks that could be used in the United States to certify roadable aircraft are the Light Sport Aircraft (LSA) certification, the Special Airworthiness Certification, and the FAR Part 23 certification. Depending on targeted applications and characteristics, one of these certifications can be used. Table 4 compares the characteristics of these three certifications. Table 4 presents regulatory trade-offs between cost and potential applications. Indeed, while LSA regulation enables a low-cost certification, the vehicle can only carry two passengers including the pilot.

**Table 4:** Comparison of currently available certifications [100, 224, 447, 448, 449]

| | **Light Sport Aircraft** | **Special Airworthiness** | **FAR Part 23** |
|---|---|---|---|
| **Maximum TOGW** | 1,430 lbs | 2,700 lbs | 19,000 lbs |
| **Maximum stall speed** | 45 kts | 61 kts | No restriction |
| **Cabin limitations** | 2 persons | 4 persons | No restriction |
| **Pressurization** | No | No | Yes |
| **Cost** | \$125-150k | \$1-5 million | \$5-50 million |
| **Commercial use** | No | No | Yes |
| **Retractable landing gear** | No | No | Yes |
| **Engine restriction** | Single reciprocating engine | No | No |

No commercial use is allowed below a certification process that costs around \$5-50 million. Aware of these limitations, the Aviation Rulemaking Committee is currently reviewing the certification process. An update of this 30-year-old certification process will revitalize current markets. Nevertheless, the changes have not been defined yet and will probably emerge between 2016 and 2020 [100]. **These doubts in future certification processes bring significant uncertainty in requirements, which will decrease over time. Hence, this evolving uncertainty in requirements needs to be considered in early design phases.**

Another element that must be taken into account while designing small personal vehicles is the ability of being piloted by a large number of people. As of today, two main pilot certifications are available: sport pilot (14 CFR Part 61, Sub-part J) and private pilot (14 CFR Part 61, Sub-part E). While the former only takes approximately 35 hours (between \$4,500 and \$6,000), the latter usually takes around 70 hours (between \$8,500 and \$10,000). However, several restrictions limit the privileges of sport pilots:

- Only Light Sport Aircraft can be piloted.

- No more than one passenger can be carried.
- No night-flying and instrument-flying allowed.
- Altitude is limited to 10,000 ft.

Hence, these restrictions must be taken into account when targeting new markets. However, for emerging vehicles, precise targeted markets are usually not fully defined and might remain unclear until later phases of the vehicle development. Nevertheless, once fully defined, regulatory requirements must be translated into design constraints. This constitutes another source of evolving requirements' uncertainty.

### 1.3.2 Possible Applications Are Not Well Defined

Revolutionary and unconventional vehicles are usually driven by technology enhancements rather than precise market-driven needs. Prototypes are then built to transform ideas into viable products while applications gradually emerge over time. Being different in nature, each application needs to be regulated by specific requirements, constraints, etc. This fuzziness in applications in early design stages also brings a high level of uncertainty in requirements and targeted objectives. Suborbital vehicles and flying cars provide two examples of such uncertain environments. Indeed, catalyzed by the Ansari X Prize, technology feasibility has been boosted by the emergence of different applications such as space tourism, satellite launch, etc. Besides, while flying cars have recently been demonstrated as technologically and economically feasible, possible applications have not been fully defined yet: transport, leisure, etc.

#### *1.3.2.1 Suborbital Vehicles*

The conversion of the idea of touristic suborbital flights into a concrete and viable experience has only occurred very recently. While Virgin Galactic is the unambiguous leader in such activity, several competitors are emerging with various concepts. The quality of suborbital flights in terms of passenger experience can be defined by five major parameters: maximum altitude reached, duration of weightlessness, duration of the flight, ticket price, and maximum load factor perceived. The first three parameters have to be maximized while

the last two have to be minimized. Table 5 compares the data advertised by the different competitors.

**Table 5:** Comparison of the various experiences [252, 457]

| Company | Maximum altitude (km) | Duration in weightlessness (min) | Flight duration (min) | Ticket price (U.S. $) | Maximum load factor (g) |
|---|---|---|---|---|---|
| **Virgin Galactic** | 110 | 3-4 | 150 | 250,000 | 5 |
| **Suborbital Corporation** | 100 | 3-5 | 360 | 102,000 | - |
| **Starchaser Industries** | ≥100 | 4 | 20 | 200,000 | 4.5 |
| **Airbus** | 100 | 3-4 | 90 | 260,000 | 3 |
| **Blue Origin** | 107 | 3 | 15 | - | 6 |
| **Rocketplane** | 100 | 3-4 | 60 | - | 4-5 |
| **Dassault Aviation** | 100 | 3 | 120 | - | 4 |

As shown in Table 5, most of the metrics significantly vary from one company to the other. Moreover, other parameters such as the type of launch, whose diversity has already been discussed in Section 1.2, and the configuration of the cabin (available space by passenger, orientation of the seats, etc.) can also be considered as key drivers of the passengers' experience and are subject to strong uncertainty. In addition to space tourism, suborbital vehicles would also support other activities. As presented by the Tauri Group [86, 435], the potential offered by suborbital flights can benefit eight different markets: commercial human spaceflight, basic and applied research, aerospace technology test and demonstration, media and public relation, education, satellite development, remote sensing, and point-to-point transportation. Hence, even if a direct application of suborbital flights seems to be space tourism, many other applications could gradually emerge and provide new requirements, constraints, and objectives.

Suborbital flights could also serve as a platform for studies related to emergency medicine in space, human behavior, biological or physical research, etc. Since a serious injury takes

no more than six minutes to become fatal, the average four to five minutes spent in micro-gravity are enough to practice, improve, and certify space emergency procedures. These flights also offer a unique environment for learning more about the alternation of micro-gravity and hyper-gravity phases. Finally, as far as other domains are concerned, these flights would allow researchers to perform their experiments on their own and without having to be well trained and physically fit.

Suborbital vehicles can also be used as a first stage to launch small satellites into Low-Earth Orbits (LEOs). Several programs have been started to study this possible application including the Orbital Suborbital Program (OSP) [383], RASCAL [154, 217, 487], and the Swiss Space Systems (S3) [374].

#### *1.3.2.2 Flying Cars*

Flying cars have generated interest since the beginning of the $20^{\text{th}}$ century. Indeed, they have been depicted in numerous movies such as Star Wars (1977-2005), Back to the Future II (1989), Blade Runner (1982), and The Fifth Element (1997). The lack of current flying cars have also become an idiomatic mark of disappointment in present technologies compared to past promises. Comedian Lewis Black said: "This new millennium sucks! It's exactly the same as the old millennium. You know why? No flying cars!" [331]. Indeed, these emblematic flying cars have the potential to revolutionize everyday life. Potential applications of these new vehicles are listed below:

- Extension of leisure General Aviation (GA) aircraft: this application seems to be the most promising in the near term. Indeed, based on current predicted prices (around $300,000), these vehicles will first be available for wealthy private pilots willing to extend the capabilities of their aircraft/car. For this market, requirements and constraints will be close to typical GA aircraft while also encompassing ground traffic regulations.

- New taxi services: flying cars can be used for faster taxi services, especially for long distances. Such commercial use will require additional safety measures and stringent certifications and regulations. Specific requirements could include higher speed and

lower cost.

- Personal door-to-door aerial vehicles: this application will require a significant decrease in cost to enable general public to afford it. In addition to this important cost reduction, solutions must be found for pilot training and air traffic control. Implementations of more sophisticated avionics and auto-pilot systems would probably be key enablers for this application.

- Strategic military vehicles: this application does not require the same requirements as the previous ones. Pilots could be highly trained and affordability is usually less emphasized. However, performance such as speed, payload, and maneuverability become critical.

Along with technological improvement of current concepts and the establishment of regulations, specific applications will start to emerge as well as a series of targeted values for the different constraints. Even if requirements' uncertainty is now extremely high, more precise values will progressively emerge for each of those constraints. **Hence, the large number of possible applications will also result in an evolving uncertainty around critical requirements such as safety, comfort, and mission definition.** All these observations lead to the following Assertion:

**ASSERTION 3: Significant uncertainties originate from customer, regulatory, and market requirements. These uncertainties, which evolve throughout the design process and as the market grows, must be addressed to support the development of robust vehicles.**

With this third Assertion, the main characteristics of emerging markets were investigated. The following section summarizes information collected in Sections 1.1 to 1.3 to help establish critical capabilities required to support the development of these emerging markets.

## *1.4 Summary*

This section intends to define the research objective by identifying specific capabilities required to ensure the development of these new markets. Finally, the main challenges that must be addressed by decision makers are also discussed.

### 1.4.1 Main Research Focus Areas

Section 1.1 showed that emerging markets are characterized by a complex multi-objective decision environment. It also highlighted the potential impact of trading these competing objectives on the design of the vehicles. In particular, it discussed the importance of considering life-cycle costs, safety, and passenger experience in the early design phases. This led us to Assertion 1: **promising future markets are characterized by a multi-objective decision space, where trade-off analyses must be conducted in early design phases, as they might highly impact the vehicles' size and configuration.** Including all potential objectives in the early design phases is crucial. This will help maximize benefits by seeking the best trade-off between price, demand, configuration of the vehicle, etc. Therefore, once the first phase of technology feasibility has been successfully achieved and a promising demand identified, life-cycle costs, safety, and other customer-oriented considerations must be integrated into the design process. These considerations will support the transition of the previous demonstrators and prototypes into new concepts that can be profitable so that both manufacturers and operators can make enough money to sustain the new market. As of today, the design process of almost all revolutionary and advanced vehicles has been following a very disorganized and cost inefficient methodology by relying on expert judgments and subjective decisions. A typical example of such inefficient programs is the American Space Shuttle [199]. Other examples can be found in Lockheed Martin's programs for which the focus has been put on performance and innovation rather than affordability. Indeed, most of the time, life-cycle costs, safety, and customer-oriented requirements have only been considered in the final phases of the design process, when the design freedom has already been greatly reduced and the major part of the costs already committed. These practices result in very expensive programs with non-affordable, unsafe,

and unattractive vehicles. Moreover, even small modifications in the design during the preliminary or the detailed design phases appear to be disastrous in terms of cost incurred. Since approximately 80% of the costs are committed by the end of the conceptual design phase, they must be integrated into the decision process no later than this phase.

Due to the complexity of aerospace systems and the large number of possible alternatives mentioned in Section 1.2, the design space for each architecture already provides a significant optimization challenge for both the vehicle and its mission. Indeed, the design and optimization of novel and unconventional concepts are characterized by a lack of relevant data, information, and knowledge that prevents the application of traditional design methods. These challenges have been identified and discussed by Mavris et al. [279]. Indeed, they suggest the creation of physics-based models and numerical simulations to alleviate the lack of historical and physical data. They also emphasize the significant role of requirements as cost drivers. Hence, there is a need to assess the impact of requirements on profitability. The evolving nature of these requirements can be addressed by shifting from deterministic, serial, and single-point designs to dynamic parametric trade environments. Finally, they also suggest the use of surrogate modeling to enable the integration of the multiple disciplines into a single environment that can support rapid parametric trade-off analyses.

Applications of the aforementioned methods will enable a paradigm shift to occur. The benefits of the latter are displayed in Figure 15 in terms of cost committed and design freedom throughout the design process. The design process is typically divided into three successive steps, which follow the requirements definition phase. The purpose of this phase is to identify and understand stakeholders' requirements. This phase is primordial since requirements enable the designers to define the problem and benefit from a good starting point for the design. With more than 22% of project failures being attributed to requirements definition, it is critical that they are properly captured [434]. The three steps of the design process characterized by an increasing level of detail of the representations and analyses are discussed below [278]:

- Conceptual design: the end-goal of this phase is the identification and selection of

the feasible and viable concepts. The number of concepts that need to be evaluated and compared is typically extremely large. This phase traditionally relies on experience, expertise, and historical data. For unconventional vehicles, when no data are available, physics-based models or approximations by surrogate models can be used. Since detailed information about the product is usually unavailable, the choice of the modeling methods is often driven by consistency and speed instead of accuracy. This phase aims at defining the overall configuration by making the main design choices and describing the general performance of the vehicle. Hence, it is a key phase as decisions have a strong impact on the overall vehicle's cost and performance.

- Preliminary design: this phase transforms the concept identified at the end of the conceptual design into a real product that can be manufactured and operated. Hence, each design variable will be sized to obtain an optimized design that meets both constraints and requirements. The concept is decomposed into several subsystems and a more detailed multidisciplinary optimization is performed. More sophisticated, complex, and accurate tools are used compared to the conceptual design phase and interactions between the different subsystems are also addressed. During this phase, designers will also face an important trade-off between model accuracy and computational time. While high-fidelity tools lead to better and more detailed results, the time spent to develop the models and run the simulation limits the number of alternatives that can be investigated. This phase ends with final decisions about the overall configuration of the vehicles.

- Detailed design: this last phase focuses on the design, development, and fabrication of every piece of the vehicle. Subsystems can still be fine-tuned but the level of freedom is highly limited. During this phase, tasks mainly rely on Computer-Aided Design (CAD) and Computer-Aided Manufacturing (CAM) software. An efficient and abundant cooperation between the different teams is also required in order to ensure a consistent design and a successful product once assembled. This phase ends with the construction of a prototype and its testing.

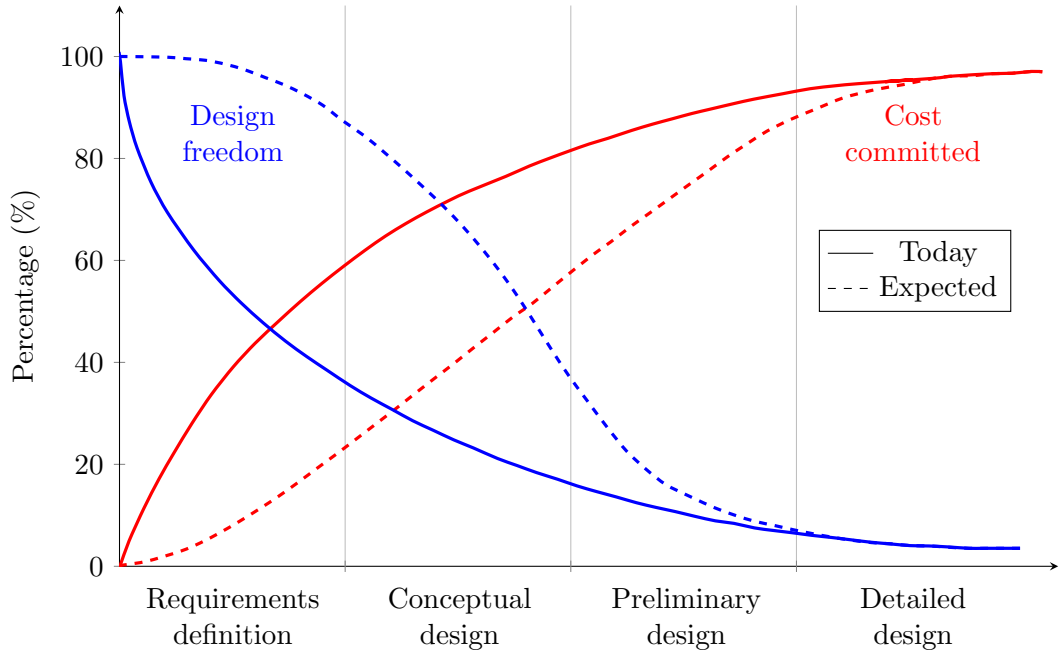



**Figure 15:** Expected benefits of the paradigm shift [277]

Hence, enabling this paradigm shift, and consequently avoiding the same mistakes to be repeated for these new vehicles, is crucial to support the development of new markets.

In addition, emerging markets are extremely challenging as they are characterized by very specific features. Firstly, Section 1.2 points out the need for a rigorous and systematic methodology able to support the selection of a design in a particularly large combinatorial design space. It also discusses the lack of baseline due to strong limitations in current design practices. In this respect, Assertion 2 was formulated: **a rigorous and systematic methodology is needed that enables the exploration of a large combinatorial design space and supports quantitative trade-off analyses to facilitate the selection of a design.** Secondly, Section 1.3 discusses the presence of high uncertainty in requirements as well as its evolution through the progressive establishment of regulations. It also shows the possible emergence of multiple applications for a given vehicle and their corresponding impacts on constraints and requirements. Hence, Assertion 3 was made: **significant uncertainties originate from customer, regulatory, and market requirements. These uncertainties, which evolve throughout the design process**

**and as the market grows, must be addressed to support the development of robust vehicles.**

As a consequence, designers and decision makers have to tackle challenging problems related to evolving requirements specific to those new markets. Due to the large number of possible architectures along with their corresponding possible configurations and missions, the architecture selection is extremely challenging. Current practices that rely on local optimization only provide good and useful results when the requirements are well known and established, and when a baseline is clearly defined. However, as discussed in Sections 1.2 and 1.3, this is not the case for those specific markets. If current practices are directly used, there is a high risk of missing hidden promising opportunities. Some combinations of features may not be considered while being promising for some set of requirements. It also provides better flexibility to designers who will be able to perform trade-off analyses across a large range of concepts. An analogy between design space and country exploration can be made. This will help understand the importance of good design space exploration techniques. To explore a new $\left|\begin{smallmatrix}\text{design space}\\ \text{country}\end{smallmatrix}\right|$, different types of $\left|\begin{smallmatrix}\text{design approaches}\\ \text{maps}\end{smallmatrix}\right|$ exist. Typical $\left|\begin{smallmatrix}\text{aircraft design approaches}\\ \text{city maps}\end{smallmatrix}\right|$ provide extremely detailed information about a very small area. $\left|\begin{smallmatrix}\text{Architecture comparison approaches}\\ \text{National maps}\end{smallmatrix}\right|$ give an overview of the entire $\left|\begin{smallmatrix}\text{design space}\\ \text{country}\end{smallmatrix}\right|$ but are not detailed enough to support important decisions about $\left|\begin{smallmatrix}\text{design choices}\\ \text{itineraries}\end{smallmatrix}\right|$. Finally, $\left|\begin{smallmatrix}\text{architecture optimization approaches}\\ \text{sets of city maps}\end{smallmatrix}\right|$ provide important information about a handful of $\left|\begin{smallmatrix}\text{architectures}\\ \text{cities}\end{smallmatrix}\right|$. These approaches are useful when $\left|\begin{smallmatrix}\text{architectures and configurations considered}\\ \text{roads and cities visited}\end{smallmatrix}\right|$ have been predefined and when there is no inherent uncertainty. However, they do not provide enough flexibility to dynamically explore the $\left|\begin{smallmatrix}\text{design space}\\ \text{country}\end{smallmatrix}\right|$ according to $\left|\begin{smallmatrix}\text{new requirements and regulations}\\ \text{weather and special event updates}\end{smallmatrix}\right|$. There is a trade-off between level of detail, area covered, and convenience. To overcome this pitfall, a $\left|\begin{smallmatrix}\text{broad design space exploration}\\ \text{GPS}\end{smallmatrix}\right|$ is needed. It will provide the required level of detail about the entire $\left|\begin{smallmatrix}\text{design space}\\ \text{country}\end{smallmatrix}\right|$ to help $\left|\begin{smallmatrix}\text{designers}\\ \text{travelers}\end{smallmatrix}\right|$ make the right decisions according to new available information. This is essential to avoid $\left|\begin{smallmatrix}\text{designers}\\ \text{travelers}\end{smallmatrix}\right|$ from wasting their time in uninteresting areas, while also allowing them not to miss some interesting areas. Baselines are often selected by experts based on previous experience in the field. However, the lack of historical data in new markets requires

more architectures described by more configurations to be considered. This broad design space exploration is the ability to consider all alternatives of the design space at a level of detail that would enable well-founded decisions to be made according to new or changing requirements. **Therefore, designers must broadly explore the design space.** In addition, they must highlight promising alternatives and identify potential technology gap(s) that would prevent the development of such markets. Moreover, the main cost drivers must be identified and the trends clearly understood. This would enable designers to emphasize certain domain(s) in terms of research and development. Finally, decision makers must be able to determine whether it is preferable to use a single baseline or several baselines for the various potential applications. Such decision requires the ability to quantify the trade-offs between performance and robustness. Hence, alternative optimization must include these two criteria as objectives by using Pareto frontiers. **This is only possible if a quantitative analysis is performed**. These aforementioned capabilities result in the formulation of the following Research Focus:

**RESEARCH FOCUS 1: To enable a broad and informed design space exploration for optimized vehicle selection at a conceptual design level.**

Lack of regulations and fuzziness in customer requirements require designers to deal with highly uncertain requirements. Besides, this uncertainty is rapidly changing throughout the establishment of the market. In addition, most probable values for these constraints will also emerge. As a consequence, current static robust design approaches rapidly face important limitations. Indeed, more flexibility is required to allow decision makers to decide if it is more profitable to start a more detailed design or to wait until more precise regulations are fixed. The risk related to each option must be quantified to support the right decision. Moreover, if designers are able to identify requirements that are the most sensitive to uncertainty, they can better negotiate the regulatory framework. Besides, more flexibility will also enable designers to update their design priorities and the corresponding requirements and constraints to match new available market analyses. **To do so, both rapid trade-offs and objective prioritization capabilities are required**. This flexibility will also

provide the crucial capability to estimate the benefits/costs of designing a single vehicle for two relatively similar applications as well as precise information about those vehicles. One of the key characteristics of emerging markets is the evolving nature of its requirements and constraints. There are two majors parameters that describe an uncertain constraint: its most probable/mean value and its degree of uncertainty. The latter is often modeled by the variance or the standard deviation of the probability distribution. As regulations and applications are emerging, these two parameters can rapidly change. The evolution of uncertainty is not similar for all constraints and consequently some requirements can converge towards fixed values faster than others. **Thus, decisions under evolving uncertainty in requirements must be supported.** This leads to the following Research Focus:

**RESEARCH FOCUS 2: To support decisions under unclear objectives and evolving uncertainty in requirements.**

The identification of these two research focuses highlights the need for a methodology that supports decision makers during the emergence of new markets. To better understand the problem and the needs, an analogy with current medical challenges is formulated below: | Designers / Physicians | deal with highly complex | new vehicles / human bodies |. Their goal is to support their | products / patients | in a world where new | regulations / diseases | are regularly appearing and evolving. In addition, there is a large amount of uncertainty in | customers' requirements / people's way of life | so that it is crucial to find | designs / drugs | robust to small changes in the aforementioned noise factors. On the one hand, a static approach to the problem, which uses very generic solutions makes them highly inefficient due to potential | competitors / virus reinforcment |. On the other hand, the development of extremely specific and customized solutions is expensive and might require detailed information that is not necessarily available on time. Hence, there is a need for more flexibility in decisions. Solutions must be adapted before the point of no return is reached. Such methodology can enable the exploration of the entire | design space / DNA | in order to avoid missing | alternatives / information | and identify new trends between | choices and results / symptoms and diseases |. Besides, it also allows | designers / physicians | to find the best solution using the maximum amount of information available at a given point in time. As a consequence, such methodology has the potential to revolutionize the

way $\left|\begin{smallmatrix}\text{vehicles are designed}\\ \text{people are treated}\end{smallmatrix}\right|$ such that more $\left|\begin{smallmatrix}\text{designs can be successful}\\ \text{people can be saved}\end{smallmatrix}\right|$ while saving a large amount of money and reducing inherent risk by using more information. Therefore, developing such competencies is crucial and will provide great benefits. The key enablers of such methodology are discussed in the following section.

### 1.4.2 Establishment of the Research Objective

Based on the aforementioned research focuses, characteristics of this new methodology are described below:

- The unusually large design space, which results from the potentially high number of viable architectures and the lack of baseline, must be explored. Not only should the methodology be able to generate and evaluate the different architectures, but it should also be capable of finding the best configuration for each architecture. Depending on the configuration, the performance of a given architecture might vary significantly due to the large number of options for each feature. Therefore, a single configuration for each architecture does not fully define or characterize an architecture and is not sufficient to support the selection of a design.

- The presence of numerous competing design objectives requires the methodology to support traceable and informed decisions enabled by quantitative trade-off analyses between objectives. In addition, once priorities have been set, the methodology must also support concept selection.

- The high uncertainty inherent to emerging markets must be addressed by designing vehicles robust to small changes in requirements and constraints. Performance of such vehicles must be quantified in order to support informed decisions.

- Due to the evolving nature of requirements' most probable values and uncertainty, decision flexibility must be emphasized through rapid trade-offs and objective prioritization. The capabilities of rapidly developing and analyzing multiple scenarios is one of the key enablers of such flexibility. To efficiently support decision makers throughout the design process, the degree of uncertainty as well as the mean value of each

constraint should also be controllable. While requirements are highly uncertain at the beginning of the design process, they are continuously freezing with the establishment of regulations and more precise applications. For that purpose, time-dependent uncertainty must be modeled and included in the decision process.

- Dominant configurations must be highlighted by identifying the least and the most affordable options for all features. Then, potential baselines can be defined and quantitatively described for higher fidelity studies and local optimization.

- The main design drivers must be identified. The impacts of each alternative on the vehicle characteristics would then be assessed. Based on a sensitivity analysis, designers must be able to decide which domains must be privileged in terms of research and attention to improve the overall affordability.

In order to develop these capabilities, the remaining of the research will be articulated around the following Research Objective:

**RESEARCH OBJECTIVE: To establish a methodology that enables a broad design space exploration at a conceptual level to select solutions against unclear objectives and under evolving uncertainty in requirements.**

To reach this objective, several methodological and technical challenges need to be addressed as discussed in the following section.

### 1.4.3 Research Challenges

The novelty of emerging markets as well as the complexity of the problem raise numerous challenges that need to be addressed for a successful development of the aforementioned capabilities.

First of all, the design of new advanced vehicles usually relies on cutting-edge technologies or is at the boundary between two established domains. For example, suborbital vehicles require technical performance similar to spacecraft, while providing operational capabilities and safety levels similar to aircraft. Flying cars must benefit from both car and

aircraft/helicopter capabilities. Hypersonic commercial transportation requires the development of new technologies for both airframe and propulsion systems in order to minimize the sonic boom, reduce the take-off noise, increase the cruise efficiency (propulsion and aerodynamics), etc. The complexity of the considered systems also points out the limitations of Cayley's design paradigm [204] which states that:

- Each function can be plainly assigned to corresponding subsystems: wing generates lift, propulsion system overcomes drag, fuselage carries payload, etc.
- The optimization of the overall vehicle can be treated as a sum of independent optimization problems for each subsystem.

These observations do not take into account the need to increase efficiency and open new flight domains that leads to highly integrated functions and subsystems. Hence, this paradigm fails to recognize that an optimized vehicle is not necessarily the sum of individually optimized subsystems, as described in Equation 2.

$$\sum \text{optimized subsystems} \neq \text{optimized vehicle} \tag{2}$$

To tackle these limitations and support efficient solution selection, a holistic approach is required that integrates all disciplines and considers their interactions.

Current guidelines and practices for conventional vehicle design might not apply to these new categories of vehicles. The novelty of such vehicles is also translated into significant lack of historical data because there is no active vehicle currently in service. As a consequence, physics-based modeling is required for an appropriate quantification of trends. Moreover, while design for affordability with early cost considerations is now commonplace in well-established industries such as aviation and automotive, emerging markets tend not to emphasize this aspect in early phases. This lack of consideration for cost combined with the fact that most aerospace companies keep their data, and especially their cost data, proprietary, makes early cost considerations another challenge.

The size of the design space defined by the various possible combinations is also particularly large due to its numerous dimensions. Indeed, in addition to traditional alternatives for

airframe and propulsion, the number of possible combinations of different types of propulsion or new technologies increases the design space. Contrary to conventional markets, which only consider one baseline concept, new markets cannot be based on qualitative considerations or predefined baselines in order to avoid the risk of missing opportunities due to the lack of knowledge in the field. All these alternatives not only enlarge the design space but implicitly include another dimension: the mission. While traditional requests for proposal usually provide a clear description of the mission that must be performed, new markets have another degree of freedom that must be addressed. Finally, requirements have not been completely defined yet and are likely to change over the next decades.

The desired methodology requires several capabilities that are not commonplace in traditional vehicle design processes. Traditionally, new designs are either market pulled or technology pushed. While the latter mainly relies on innovative ideas and technologies that try to be transformed into a marketable product, the former directly comes from an interpretation of a request for proposal. While different architectures might be considered through morphological matrices, a baseline is always selected for more detailed trade-offs. This baseline is often selected based on experts' judgment and previous experience in this field. However, the novelty of the considered vehicles, their large variety of possible configurations, and the lack of predefined baseline require the methodology to explore the design space more deeply. Hence, more architectures and more configurations per architecture must be evaluated in order to support decisions. This involves the use of an optimization process that must be able to take architectures defined by different variables as inputs. The complexity of the problem also increases with the hybrid nature of the design variables, which can be continuous, discrete, and categorical.

To address these challenges and meet the Research Objective, new capabilities need to be developed, as described in the following section.

### 1.4.4 Required Capabilities

In order to support an improved design space exploration, the proposed methodology must be able to find solutions that are better than the ones found using existing methodologies. In addition, due to the lack of baseline and the numerous possible combinations of technologies, the methodology must be able to consider multiple concept configurations evaluated by different modeling and simulation environments. Because the design spaces of interest are usually extremely large, the methodology must be efficient and fast to compute, while also providing the ability to capture important trends and perform key trade-off analyses.

To support decision-making under evolving uncertainty in requirements, the methodology must handle the dynamic behavior of uncertainty. Go/no-go decisions must also be supported with quantitative and analytic analyses. Besides, the methodology has to support the rapid development of various scenarios in order to assess the risk and the expected performance related to each potential decision.

Finally, in order to provide insights on the market of interest, the methodology has to help decision makers identify the key baseline(s) in the multi-objective solution space, along with a detailed description of these baselines.

To help the development of this methodology, the following chapter reviews, compares, and evaluates existing methods and approaches. By identifying potential gaps that must be bridged, it will help select the different methods, tools, and approaches that can be leveraged to develop the required capabilities.

# CHAPTER II

# PROBLEM DEFINITION

This chapter presents a review of the state-of-the-art in fields relevant to this research. It will help understand the current practices along with their assumptions, strengths, and limitations. The purpose is either to find the best available methods/tools or to identify the potential gaps that could prevent the previous objectives from being met. Hence, this chapter will be articulated around several Research Questions originating from breaches in current practices that must be addressed. A description, comparison, and evaluation of existing techniques will help develop Hypotheses, which correspond to proposed solutions to the aforementioned Research Questions. This chapter starts by discussing the first Research Focus: **to enable a broad and informed design space exploration for optimized vehicle selection at a conceptual design level.** Then, it addresses the second Research Focus: **to support decisions under unclear objectives and evolving uncertainty in requirements.**

## *2.1 Design Space Exploration*

As mentioned in Section 1.4, a successful completion of the Research Objectives requires a methodology with design space exploration capabilities that are not commonplace in traditional aerospace conceptual design approaches.

### 2.1.1 Gap Identification

Three different design approaches have been identified in the literature: typical aircraft design process, architecture comparison, and architecture optimization. New aircraft designs usually originate from either a market pull or a technology push. To match these two market penetration models, two approaches are usually used and are described by Mavris et al.: Technology Impact Forecasting (TIF) and Technology Identification Evaluation Selection (TIES) [236, 276, 280, 284, 403]. TIF is a top-down approach that seeks the best

combination of technologies to meet future goals. The TIF methodology is described as a six-step process [403]: define the modeling environment, define the baseline, define variables and responses, create surrogate models, define the technology scenarios, and conduct the probabilistic simulation. TIES is a bottom-up approach that forecasts and discusses the ability of a combination of technologies to meet future goals. The TIES methodology is described as a nine-step process [276]: define the problem, identify a baseline and the configurations that will be examined, define the modeling and simulation environment, explore the design space, determine the viability of each concept, define technologies that might be infused, evaluate the impact of each technology on the vehicle, populate the evaluation matrix, and select technologies. These two approaches are based around a predefined baseline usually selected by experts thanks to their experience in the field. Besides, new concept ideas or technology innovations are brought by subsystems on existing baselines in order to reduce the overall uncertainty and the risks associated with new designs. A direct consequence of such practices on the market is the increase of the number of derivative aircraft within generations and the re-use of successful technologies between families [158]. Hence, current aircraft design practices only consist in local design space explorations for new technology infusion. The same observations can be made about the approach used to design launch vehicles, which optimizes a baseline to meet future requirements. This approach is described by several researches [4, 25, 72, 93] and requires important technology choices to be made in order to set the optimization process. In particular, Collange et al. [93] propose a six-step process for the conceptual design of future launch vehicles: selection of technologies, preliminary staging based on simplified calculations, propulsion system design, sizing of the different stages, trajectory and performance optimization, and loop to reach an optimized and feasible launch vehicle. Hence, the typical approach for launch vehicle design follows the same principles as the typical aircraft design approach and consequently provides the same advantages and drawbacks.

Newer aerospace applications such as hypersonic aircraft and interplanetary spacecraft are more likely to see breakthrough programs with new architectures. Hence, a wide range of research has been conducted in terms of architecture selection. Some methods require

designers to manually input the different architecture alternatives [454, 455] in order to perform the concept optimization. Others describe a systematic way to generate and compare architectures, but do not allow architecture optimization [18, 117, 153, 242, 346, 468]. Consequently, if newer applications were to favor a wider design space exploration, none of the current methods would be able to systematically cover the entire design space. Figure 16 shows the ability of current design approaches to explore the design space.

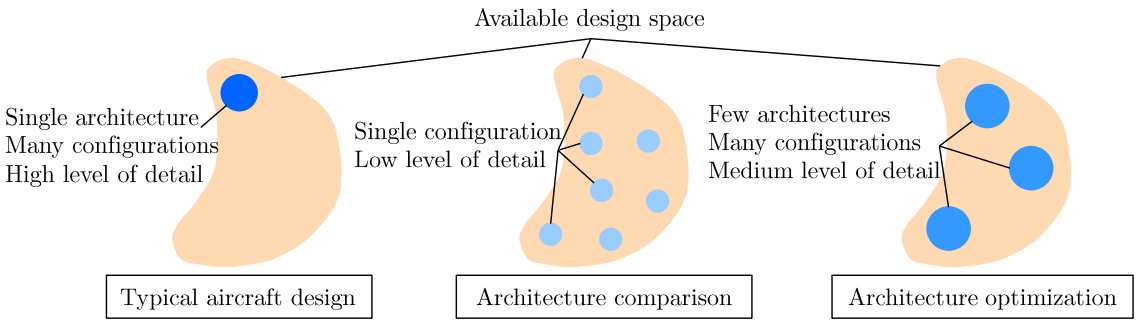



**Figure 16:** Performance of current aerospace design processes

The entire available design space is represented in beige. The blue circles represent the areas of this design space covered by the corresponding design approach. The size of the circles represents the amount of configurations considered by a given design process: the larger the circle, the larger the number of design variables. The intensity of the color represents the level of detail of the associated modeling and simulation framework. The typical aircraft design approach considers few configurations around a single baseline architecture. The corresponding analysis framework is extremely accurate and detailed. This leads to a single dark circle with a relatively large size. The architecture comparison approach compares many architectures but the level of detail of the evaluation framework is often based on qualitative information. In addition, only one configuration per architecture is considered, and there is no optimization. This results in numerous small light blue circles. The architecture optimization approach considers fewer baseline architectures and performs separate optimizations before comparing the best configurations from each architecture. While numerous configurations are considered for each architecture, the modeling environment is

more accurate than the architecture comparison approach but less accurate than the typical aircraft design approach. This leads to two to three large and relatively light blue circles.

To illustrate the limitations of current design approaches, their application to the design of suborbital vehicles is investigated. Typical aircraft design approaches, which aim at optimizing a predefined architecture, have been widely used and have conducted to the design of many different optimized configurations, as described below:

- Flittie et al. designed a VTVL vehicle powered by a hybrid propulsion system [149]. Designers from the American Rocket Company, Martin Marietta Manned Space Systems, and United Technologies Chemical Systems Division wanted to demonstrate the feasibility and viability of large hybrid boosters.

- Haigneré et al. optimized an air launched vehicle with horizontal landing capabilities [190]. Their goal was to develop an extremely safe vehicle using on the shelf technologies.

- Frank et al. optimized an Horizontal Take-Off and Horizontal Landing (HTHL) concept based on a hybrid rocket propulsion system with auxiliary turbofan engines [157]. The architecture was selected using qualitative considerations and experience from the team.

- Chong et al. developed another HTHL concept powered by a Rocket-Based Combined Cycle (RBCC) engine [177]. They used fixed requirements and a predefined baseline to perform a multidisciplinary optimization.

- Chavagnac et al. developed a winged bizjet-sized vehicle powered by both turbofans and rocket engines [77]. The optimization of the configuration was made around a baseline defined by the authors' expertise to meet well-defined requirements.

All of these designers claim to have developed the best vehicle according to their specific set of requirements. However, they did not explore other architectures. Hence, they might have missed some promising architectures due to the extensive use of expert judgment and qualitative considerations to select their baseline. In addition, they did not consider

robustness in their design or even assess the performance of their vehicles with respect to small changes in requirements. Often, the level of detail considered for performance and alternative evaluation meets, or even exceeds, the needs for conceptual design decisions. **Hence, while benefiting from good modeling capabilities, the typical aircraft design approach only covers a tiny part of the design space.**

The one analysis that uses an architecture comparison approach has been conducted by Sarigul-Klijn et al. [377]. They decomposed the vehicle and the mission into several features such as take-off mode, landing mode, airframe configuration, and propulsion system. They used first principle equations as well as comparison with historical data to characterize each alternative. No optimization was performed and the analysis was based on precise requirements originating from the Ansari X Prize. They suggested four promising architectures as potential winners of the Ansari X Prize:

1. Vertical take-off, aerodynamic decelerator
2. Vertical take-off, wings with wheel landing
3. Some air launch, aerodynamic decelerator
4. Some air launch, wings with wheel landing

Based on these four promising architectures, they performed a thorough study considering safety, affordability, and customer acceptance. They found that the preferred configuration for suborbital space tourism is a Vertical Take-off and Horizontal Landing (VTHL) concept powered by hybrid rocket engines. However, no detailed information has been provided on either the vehicle configuration or its mission. First principle considerations cannot be used to successfully evaluate and compare each alternative. **While considering more alternatives than typical design approaches, architecture comparison approaches do not provide enough information to support informed decisions.**

Finally, some methodologies have been developed to optimize different alternatives of suborbital vehicles but are restricted to a small part of the design space. Indeed, Huang et al. created a methodology to optimize HTHL concepts in terms of propulsion, trajectory,

etc. [212, 213, 214]. They discretized design variables to generate cases to be evaluated by the analysis code. This methodology is similar to a small-size Design of Experiments (DoE) performed on a subset of possible alternatives. Villeneuve et al. developed a methodology for architecture selection of launch vehicles [455]. While it can easily be applied to suborbital vehicles, this methodology only optimizes a handful (3-4) of architectures. **Thus, these methods cannot be used to systematically cover the entire design space because of the lack of centralized optimization process.**

The evaluation of the capabilities of current aerospace design processes shows a lack of methodology able to systematically cover the entire design space, as required to meet the Research Objective. Therefore, observations from the review of existing design approaches give rise to the first Research Question:

**RESEARCH QUESTION 1: How can current conceptual design approaches be improved to enable a broader exploration of large and complex design spaces?**

To address this question, capabilities of existing approaches are compared in Table 6. The desired methodology requires all architectures to be considered to avoid any missing opportunities. Moreover, it also requires the configurations of each architecture to be optimized since a given architecture can be described by a large number of configurations.

**Table 6:** Comparison of the various design space exploration methods

| | **All architectures** | **Generalized selection** | **Configuration optimization** |
|---|---|---|---|
| **Typical design** | | | ✓✓ |
| **Architecture comparison** | ✓✓ | ✓✓ | |
| **Architecture optimization** | ✓ | | ✓✓ |

Table 6 shows that a combination of the architecture comparison approach and the architecture optimization approach will benefit from all required capabilities. Indeed, such

methodology will be able to consider and compare as many architectures as architecture comparison methods by using their capability to systematically generate alternatives. This capability is translated into a large number of circles in the design space. The methodology will also be able to optimize configurations within each architecture by considering multiple variables for each subsystem similar to architecture optimization methods. This capability corresponds to larger circles (Figure 17). Finally, in order for the impacts of each design choice to be captured, an accurate modeling and simulation environment must be implemented through physics-based modeling. This last capability is translated into darker circles. Figure 17 represents the desired performance of the proposed methodology in terms of design space exploration capabilities.

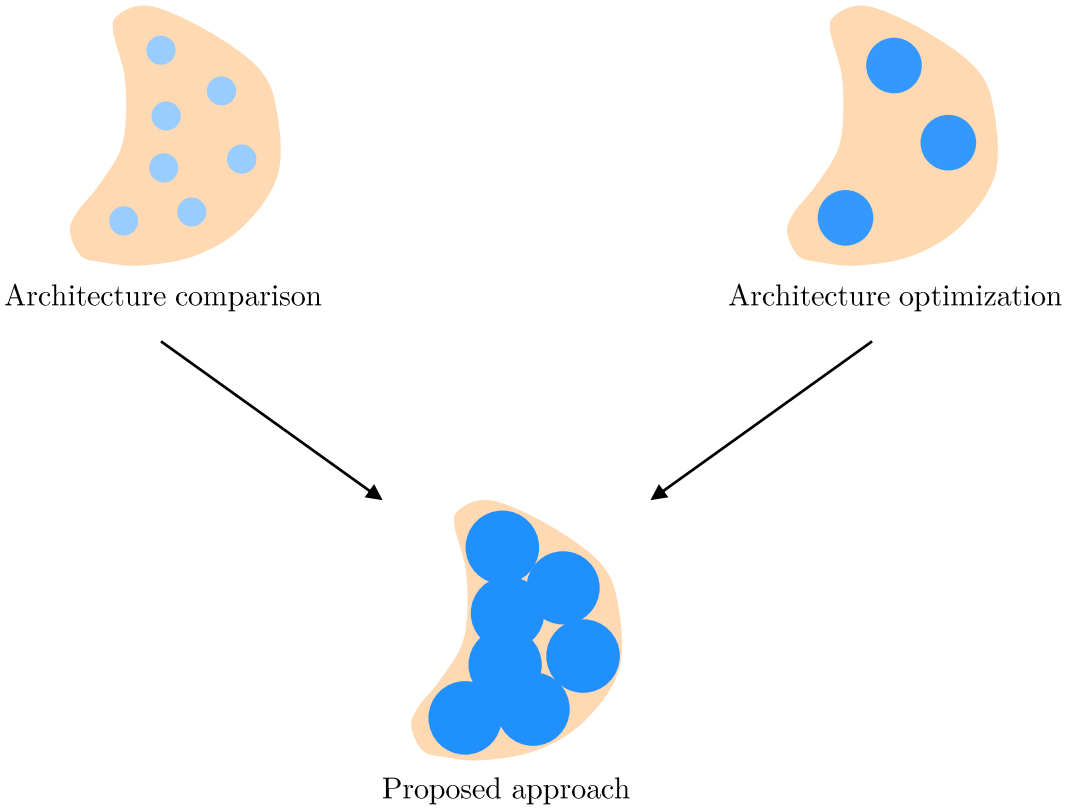



**Figure 17:** Expected capabilities of the proposed methodology

However, the same challenges that limit existing approaches also prevent such methodology to be implemented. Indeed, the various alternatives generated by the architecture comparison approach are not defined by the same variables and do not necessarily require the same modeling and simulation environments or equations to be optimized. For example, flying cars based on wings and propellers cannot be optimized with the same variables and the same modeling and simulation environment as the ones that are based on helicopters. Hence, a single centralized algorithm that can both optimize and compare alternatives cannot be defined because the design variables and the modeling and simulation environment are different. This is the fundamental flaw in current processes that explains the limitations of the aforementioned methods:

- Approaches that can compare all architectures are based on weak optimization processes. Indeed, only few generic variables common to all architectures are considered. However, these variables are not precise enough to capture trends within each architecture and do not support their optimization.

- Approaches that optimize architectures are restricted to a subset of architectures defined by the same variables. Trade-offs between variety/number of architectures and level of detail of their design variables ultimately appear. This rapidly limits the scope of such methods.

To address this challenge, the methodology needs to systematically generate alternatives that can be further optimized and compared. This requires the ability to develop an algorithm that can both optimize alternatives that are not defined by the same design variables and/or evaluated by the same modeling and simulation environment, and systematically compare them. Since performance evaluation of such alternatives might be long, the algorithm needs to be efficient. This excludes full-factorial approaches or large DoEs. The creation of a methodology that integrates both architecture comparison and architecture optimization into a single process can be broken down into two steps. The first step consists in the systematic generation of alternatives while the second step aims at comparing

and optimizing the previously generated alternatives. The remaining of this section sequentially addresses these two steps by analyzing current practices to help develop the new methodology.

### 2.1.2 Systematic Generation of All Feasible Alternatives

Even though it might appear common in aerospace, the first step hides several challenges that prevent current practices to be used. Indeed, alternatives that have to be evaluated are not predefined. Therefore, the generation of alternatives needs to go beyond a selection among existing concepts by systematically generating all possible alternatives. In addition, alternatives must be generated in such a way that they can be further optimized and compared. Hence, this alternative generation must include design variables. Indeed, in order for an optimizer to optimize the different alternatives previously generated, they need to be described by the same variables. This "apple to apple" comparison is a requirement of this architecture generation process. Finally, since optimization processes might be long to run, it is preferable to reduce the number of algorithms to be executed. These challenges lead to the following Research Question:

**RESEARCH QUESTION 1.1: How can we systematically capture all feasible alternatives, which are not necessarily defined by the same design variables, for further comparison and optimization?**

To address this question, existing alternative generation methodologies are compared and evaluated. They will then be leveraged to develop a new methodology that benefits from the aforementioned capabilities.

#### *2.1.2.1 Review of Existing Alternative Generation Methods*

First, existing alternative generation approaches are reviewed in order to be later leveraged.

**6-3-5 method:** Created by B. Rohrbach in 1968, this method aims at generating 108 new ideas in half an hour [259, 327]. The method requires six designers who sit together

and each participant has to write three ideas every five minutes. Ideas are then passed to the next designer who uses it for inspiration to create more ideas. Hence, after six rounds, a total of 108 ideas are generated. This method encourages quantity without ensuring quality. Moreover, there is no rigorous or systematic process involved and the ideas rely on experience of the designing team.

**Design Catalogues:** This method is usually used by selecting a handful of representative solutions instead of all possible ones [370]. It is very popular in scenario development. However, it only provides a limited number of alternatives to be considered for further analyses. For example, in the context of oil price forecast, instead of considering billions of combinations of price scenarios, only three are considered: baseline, optimistic, and pessimistic. Using this method, designers will miss some opportunities and will not be able to screen the entire design space. In addition, the creation of these representative solutions requires experience in the field, which is usually extremely limited for new markets.

**Theory of Inventive Problem Solving (TRIZ):** Created by Altshuller, this theory relies on the exploration of past solutions [164]. Indeed, Altshuller categorized problems into five levels with respect to their degree of inventiveness and defined their occurrence: apparent solution (32%), minor improvement (45%), major improvement (18%), new concepts (4%), and discovery (1%). As a consequence, he developed a methodology that would work for about 95% of the problems using a five-step process:

1. Identify the problem
2. Formulate the problem by identifying potential gaps or technical difficulties
3. Search for previously well-solved problems
4. Identify analogous problems with known solutions
5. Adapt identified solutions to the stated problem

However, this methodology does not work for innovative concepts and lacks of rigor in the generation of concepts.

**Morphological Analysis:** The General Morphological Analysis (GMA), developed by Zwicky, is a methodology to explore all possible solutions in multi-dimensional and non-quantifiable complex problems [498, 499, 500]. One of its most common forms is the morphological matrix used in numerous aerospace design methodologies [160, 207, 276, 403]. The system is decomposed into functions or features (rows in the morphological matrix) for which different alternatives are identified (columns in the morphological matrix). Then, solutions are created by generating all combinations of alternatives. This full-factorial approach to design space definition is rigorous and systematic but also generates non-feasible combinations. Hence, this method is usually combined with a compatibility matrix [161]. While the combination of the morphological matrix with the compatibility matrix will remove infeasible solutions, the number of alternatives to be considered usually remains extremely large.

**Interactive Reconfigurable Matrix of Alternatives (IRMA):** The aforementioned morphological matrix has been improved to account for the dynamic and iterative nature of decision-making processes [136, 221, 273, 341]. Indeed, the Interactive Reconfigurable Matrix of Alternatives (IRMA) embodies both the morphological and the cross-consistency matrices within a single software framework. It benefits from the following advantages: hidden integrated compatibility matrices, filters that can reduce the number of alternatives, standardized and flexible, integrated method for alternatives selection, etc. However, it does not track design variables and the number of possible solutions generated is still extremely large.

**Adaptive Reconfigurable Matrix of Alternatives (ARM):** Even though the IRMA greatly supports the generation of design concepts, it is based on a static functional decomposition. To overcome this pitfall, Adaptive Reconfigurable Matrix of Alternatives (ARM) [18] has been developed that relies on functional induction. It acts as a hybrid of the IRMA and the function/means tree so that both functional and physical breakdowns can be interactively managed. While being more flexible than the IRMA, it suffers from the same main pitfall: design variables are not tracked. In addition, this tool aims at

supporting the designers in the concept selection rather than in the concept generation. The ARM has been implemented into the Architecture Design Environment (ADEN) [18], whose goals were to manage complex relationships between architectural elements and to provide a framework for trade-off analyses and performance evaluation of a single generated alternative.

**Decision tree:** Decision trees map out all possible paths that can be followed to achieve a primary goal and its corresponding sub-goals [233]. Hence, if goals are linked to vehicle features, decision trees can help lay out alternatives. This approach is systematic and logic and thus reduces the probability to omit items. Compatibility issues are directly addressed so that only feasible solutions are generated at the end nodes. This approach is similar to the morphological matrix except that it requires designers to individually generate all alternatives. Hence, even if it does not require a separate compatibility matrix, the same amount of thinking is needed from the designers. However, it is less rigorous and systematic than the morphological matrix since designers must manually build each branch of the tree diagram. Finally, this method does not track design variables either.

**Morphological Evaluation Machine and Interactive Conceptualizer (MEMIC):** Morphological Evaluation Machine and Interactive Conceptualizer (MEMIC) [19] is an automated concept generator able to produce multiple design solutions from a given set of sub-functions. It relies on a web-based repository. The tool uses function-component relationships embedded in a function-component matrix and the component-component compatibility contained in a Design Structure Matrix (DSM). This tool aims at assisting designers in the choice of the various options for each function and sub-function while also supporting the generation of several "good" solutions. However, it does not include a systematic generation of alternatives in order to cover the entire design space and does not capture the design variables that define the different alternatives.

Based on the previous analysis, the best way to explore as many alternatives as possible is to decompose the system into features. In particular, it appears that the combination of

both morphological and compatibility matrices represents the best solution to systematically and rigorously generate all feasible alternatives. However, the number of alternatives that must be considered for further optimization and comparison is extremely large. Since the multi-objective optimization of each concept requires a significant amount of setup and execution time, it becomes rapidly infeasible to simply use this process. In addition, this process does not consider the downstream use of the generated alternatives. In this research, they must be further used by an optimization algorithm to be compared and optimized. In order to consistently use existing optimization algorithms, alternatives must be defined by the same variables, which rarely happens when dealing with unconventional concepts. Thus, while a sequential use of conventional morphological and compatibility matrices enables a systematic generation of alternatives, it is not conducive to further comparison and optimization. These challenges are addressed in the following section with the development of an improved morphological analysis based around variable-oriented architectures.

#### *2.1.2.2 Proposed Process*

If the generated alternatives have to be further considered within an optimization algorithm, one must ensure that they are described by the same design variables. For example, a winged body does not have the same design variables as a slender body and cannot be optimized by the same algorithm. To overcome this challenge, groups of alternatives that are defined by the same variables can be created. This will reduce the number of multi-objective optimization algorithms that must be individually launched, while also providing consistent information to these algorithms. To do so, and consequently improve the conventional morphological matrix, some terms need to be clearly defined:

- Alternative: given set of design variables sufficient to fully define a concept
- Features: functions or physical elements that can vary among alternatives
- Options: various available choices for a given feature
- Architecture: group of alternatives that are described by the same design variables

- Configuration: given set of design variables that freeze the design of a given architecture

Based on these "new" definitions, a two-step process is proposed (Figure 18). The first step consists in grouping the different options that can be described by the same design variables. This reduces the number of available options for each feature and consequently the number of possible architectures. The design variables used to describe and characterized the different options of a given group are included in the optimization process. As a consequence, the number of possible alternatives is increased but the number of discrete architectures is decreased. Instead of decreasing the design space, continuous variables are kept and consequently an infinite number of potential alternatives are considered. This will make the following optimization process more efficient. The second step aims at removing the features that are only described by one group of options in the morphological matrix. Indeed, instead of decomposing this feature into several options, its specific set of design variables is used in the optimization process.

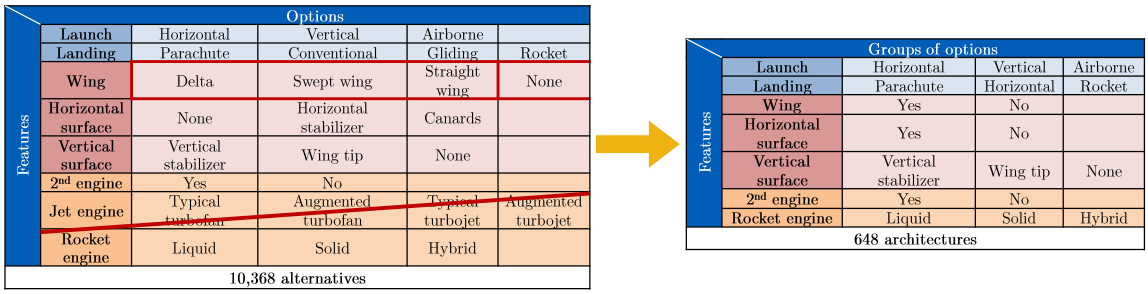



**Figure 18:** Improvement of the morphological analysis

Following this two-step improvement of the traditional morphological analysis, a typical compatibility matrix is created to ensure that only feasible alternatives are generated.

This new process provides significant benefits to the development of the methodology discussed in Section 1.4. Indeed, it provides a systematic and rigorous methodology that is able to capture all alternatives. Compared to the traditional approach, it decreases the number of optimization algorithms that must be executed and thus decreases set-up and

computation times. Hence, the new process provides a methodology to generate alternatives in order to further foster an efficient optimization and comparison of all possible concepts. This leads to the following Hypothesis:

**HYPOTHESIS 1.1: IF a variable-oriented morphological analysis combined with a compatibility analysis is developed THEN all feasible alternatives can be systematically generated for further comparison and optimization.**

In order to validate this Hypothesis, Experiment 1.1 is implemented. First, a list of all possible options for each architecture needs to be created using knowledge gained from a literature review. Next, a conventional morphological matrix will be created and modified based on the previous methodology and the new "architecture" definition. Finally, a compatibility analysis will be developed and combined with the previous variable-oriented morphological analysis. To fully validate Hypothesis 1.1, three elements need to be validated: exhaustiveness, feasibility, and ability to be further compared and optimized. In particular, the number of discrete optimization algorithms to be executed has to be greatly reduced in order to foster large design space explorations. According to various studies, the execution time for a single optimization algorithm is around several days [50, 52, 325]. As a consequence, reducing the number of algorithms is a key enabler of large-scale design space exploration. As described in Section 1.2, typical conceptual design spaces for advanced vehicles are composed of at least 500,000 alternatives. So, in order for such exploration to be manageable, the number of discrete optimization algorithms must be reduced by a factor $10^5$. This would result in an overall computational time of about one week. The validation of Hypothesis 1.1 will be addressed through the following four questions:

1. Are all existing concepts covered by the final set of alternatives?
2. Are all alternatives feasible?
3. Is the number of discrete optimization algorithms to be executed greatly reduced compared to the typical morphological analysis?

4. Is each architecture defined by a unique set of design variables?

Once all the alternatives have been generated by the aforementioned process, they must be optimized against stated criteria and compared to enable designers to make informed decisions. This is discussed in the following section.

### 2.1.3 Comparison and Optimization of Alternatives

The goal of this section is to provide a methodology that can efficiently and simultaneously compare and optimize the generated alternatives against multiple criteria. To reach this goal, the following challenges need to be addressed:

- Variables are described by design variables that can be continuous (sweep angle), discrete (number of engines), and categorical (type of propellant). Therefore, conventional efficient gradient-based algorithms cannot be used.

- Architectures can be defined by different design variables and evaluated with different modeling and simulation environments. Hence, the use of a single conventional optimization algorithm is impossible.

- There are potentially many architectures to be optimized and many discrete or categorical design variables. Consequently, surrogate models cannot be used to speed up the process.

- There are multiple objectives that are not prioritized. As a consequence, multi-objective optimization algorithms must be used. However, due to the lack of a priori prioritization and required capabilities to visualize trends, aggregation of the multiple objectives into a single objective function is not possible.

These challenges require the development of a new optimization process. This leads to the following Research Question:

**RESEARCH QUESTION 1.2: How can the Pareto frontier of solutions defined by different sets of design variables be efficiently determined in a design space composed of discrete, continuous, and categorical variables?**

In order to address this question, the first step is to optimize each architecture with respect to its specific design variables. To do so, existing multi-objective optimization algorithms are described, compared, and evaluated to support the development of the new process. Only algorithms that can, at least, deal with continuous variables are presented. Their ability to also handle discrete, and consequently categorical, variables will be evaluated. A typical formulation of such problem is presented in Equation 3, where $y$ is the vector of objectives and $x$ the vector of variables. $g$ and $h$ represent the inequality and equality constraints, respectively.

$$\begin{aligned} \min_x \; & y = f(x) \\ & g(x) \leq 0 \\ & h(x) = 0 \end{aligned} \tag{3}$$

To solve this problem, two main approaches can be taken. The first approach called compromise optimization (or a priori multi-objective optimization) defines an Overall Evaluation Criterion (OEC), which is a value function that aggregates multiple attributes. In this case, the user preferences are included upfront. The second one called Pareto optimization (or a posteriori multi-objective optimization) generates a set of optimized solutions which represent relative optimality between attributes. In this case, the user preferences are captured a posteriori.

#### *2.1.3.1 Compromise Programming*

One of the first ideas when dealing with multiple objective is to combine them into a single aggregate function using weights to emphasize some of the objectives. For example, minimizing the fuel consumption could be preferred to maximizing the cruise speed. Even if a multitude of formulations exists, one of the most widely used is the one suggested by G. N. Vanderplaats [451] and presented in Equation 4, where $w_k$ is the weighting factor corresponding to the k$^{\text{th}}$ objective function $f_k$, $f^\star$ is the k$^{\text{th}}$ function target, and $f^-$ the k$^{\text{th}}$ worst known value of the k$^{\text{th}}$ objective function.

$$f(x) = \left\{ \sum_k \left[ w_k \frac{f_k(x) - f_k^\star(x)}{f_k^-(x) - f_k^\star(x)} \right]^2 \right\}^{0.5} \tag{4}$$

Another formulation was provided by Mavris et al. [274] within the aerospace domain to support the selection and evaluation of military aircraft. The OEC developed is presented in Equation 5. In this formulation, the Greek letters correspond to the weighting coefficients and each ratio represents a criterion: Life-Cycle Cost (LCC), Mission Capability Index (MCI), Engine-related Attrition Index (EAI), Survivability Probability (SP), and Operational Readiness (OR). For criteria that must be maximized, the baseline denoted by the subscript $BL$ is the denominator and for criteria that must be minimized, the baseline is the numerator.

$$OEC = \alpha\left(\frac{LLC_{BL}}{LLC}\right) + \beta\left(\frac{MCI}{MCI_{BL}}\right) + \gamma\left(\frac{EAI}{EAI_{BL}}\right) + \delta\left(\frac{SP}{SP_{BL}}\right) + \epsilon\left(\frac{OR}{OR_{BL}}\right) \quad (5)$$

While being fast and easy to implement, these methods are very subjective because they depend on the definition of weighting factors. Moreover, some formulations could be very sensitive, especially if extremely small and/or high values are used. Similarly, criteria with a broader range could dominate the optimization. To overcome these drawbacks, another more complex approach has been developed, as discussed in the next section.

#### *2.1.3.2 Pareto Optimization*

Instead of seeking for one optimum solution, the idea is to generate a set of optimum solutions. For that purpose, the Pareto optimization is based on a partial ordering space instead of a total ordering space. This space has the following rules [170]:

- A "weakly dominates" B if A is better in some attributes and equal in others.
- C "strongly dominates" B if C is better in all attributes.
- A and C are "incomparable" if A is better than C in some attributes but worse in others.

Based on these rules, the subset of all non-dominated points is called the Pareto frontier (Figure 19). Even though this technique does not provide a single optimized point, the visualization of the Pareto frontier allows decision makers to better understand trade-offs that must be made. It then helps them formulate a single-objective optimization problem

to select their "best" design. This advantage becomes even more important when uncertainty is present in requirements. This technique is harder and more complex to implement but is promising and has already provided significant benefits in the conceptual design of spacecraft, launch vehicles, and supersonic aircraft [65, 87, 289, 455]. Another important advantage of this approach is its applicability to sub-problem optimization with its capability to only keep non-dominated alternatives without any decisions.

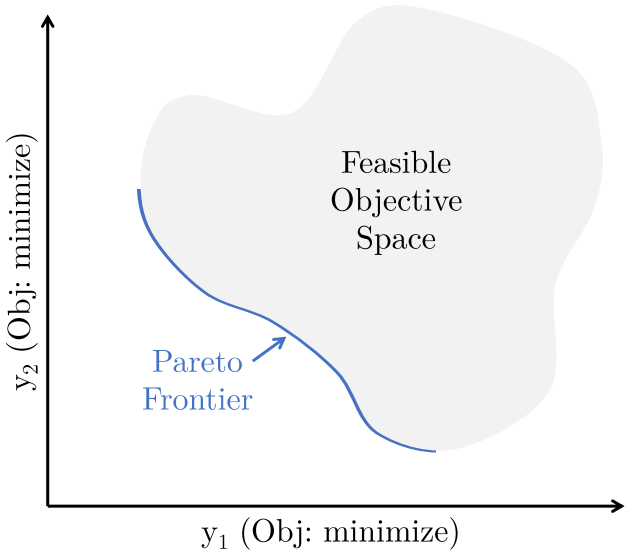



**Figure 19:** Notional example of a two-objective Pareto frontier [170]

Determining the Pareto frontier is not an easy task, especially when the number of criteria, constraints, and variables is very large. Two major strategies of Pareto frontier sampling can be identified [170].

The first strategy consists in defining and solving a series of single-objective problems so that each of them provides a point on the Pareto frontier. This can be achieved by varying either the objective function or the constraints. A common approach is to define a parametric objective function as the weighted p-norm presented in Equation 6, where $p$ is the selected norm (usually 1 or 2), $y_i$ the different objectives, and $w_i$ their corresponding weighting factors that have to be changed for each sub-problem.

$$f(x) = \left( \sum_i w_i y_i^p(x) \right)^{1/p} \tag{6}$$

Other more sophisticated methods based on the same operating principle also exist such as the Epsilon-Constraint method [191], the Normal Boundary Intersection (NBI) method [104], and the Normalized Normal Constraint (NNC) method [293]. While being fast, easy to implement and relatively straightforward, these techniques do not work well for non-convex, non-linear, and highly constrained design spaces.

The second strategy consists in using evolutionary algorithms. They approach the Pareto frontier by iteratively optimizing a population of points. Ghosh and Dehuri [173], Tan et al. [426], as well as Foncesca and Fleming [151] provide a survey of such existing algorithms. A brief description of each of them is provided along with their main advantages and drawbacks in order to help determine the most suitable algorithm for the stated problem.

- Weighted-Sum-Approach (WSA): it uses a genetic algorithm to generate the weights $w_i$ of the objective function presented in Equation 6. While being computationally efficient, this approach has some difficulties to find accurate weighting factors when there is a scaling disparity between objectives. Moreover, it cannot handle non-convex design space.

- Vector Evaluated Genetic Algorithm (VEGA): it is an extension of the simple genetic algorithm since it only modifies the selection phase. Indeed, it creates sub-populations generated from the old generation by a proportional selection for each objective. Then, it merges these sub-populations to create the next generation. This approach is simple and easy to implement but has some issues when dealing with non-convex design spaces.

- Niched Pareto Genetic Algorithm (NPGA): it mainly relies on a modified tournament selection, which includes the dominance principle among the selection criteria. This approach is complicated to implement and its performance highly relies on the tournament size and the sharing factor among two populations. Nevertheless, it is very fast to run.

- Non-dominated Sorting Genetic Algorithm (NSGA): it uses the same operating principles as the genetic algorithm (i.e. reproduction, mutation, and selection). It classifies the individuals into several layers based on their fitness. This fitness is attributed based on the "level of dominance". This algorithm is relatively easy to implement but is very sensitive to initial parameters. To correct this pitfall, an improved version called Non-dominated Sorting Genetic Algorithm-II (NSGA-II) has been created in 2002 [110]. This version takes into account the crowding distance, which is a measure of the relative distance between points. It enables an evenly-spaced sampling of the Pareto frontier and is also very efficient when dealing with unconventionally shaped design spaces. Even if this approach is more resource and time-consuming, it has the capability of handling all kinds of design spaces and complex problems.

- Multi-Objective Genetic Algorithm (MOGA): it uses non-dominance and determines its level by assigning a value to each individual compared to all the others. It is different from the NSGA, which proceeds by layers of non-domination. This algorithm is efficient and fast to run but the performance highly depends on the sharing factor. This sharing factor is used to weight the diversity in fitness assignment [90].

- Strength Pareto Evolutionary Algorithm (SPEA): it combines the notions of elitism and Pareto non-dominance. Indeed, at every generation, a set of non-dominated points is kept from the previous generation. Elitism is then used to update the population and the set of non-dominated points at each generation. Contrary to the MOGA or NPGA, this algorithm does not use any predefined parameters so that it is more robust to the nature of the problem and to the initial settings. Nevertheless, it is relatively inefficient for non-convex spaces.

- Predator-Prey Evolution Strategy (PPES): relatively different from the other algorithms, it uses the concept of predator-prey to assign fitness to individuals. An undirected connected graph is built and preys are placed on its vertices. Then, predators (one per objective function) are located on the graph and catches the prey with the

best value according to its criterion. Predators move in the graph until the convergence criterion is reached. The method is simple and easy to implement but it often results in unevenly-spaced Pareto frontiers and is very sensitive to predefined parameters.

- Thermodynamic Genetic Algorithm (TGA): it evaluates the fitness of each individual by a function related to thermodynamic equilibrium called Gibbs energy. It enables the convergence towards the Pareto frontier while preserving a good diversity among the obtained solutions. The algorithm is based on a predefined annealing schedule and thus the overall performance of the algorithm depends on the definition of this schedule.

#### *2.1.3.3 Tool Selection for Architecture Optimization*

To select the most suitable method, it is important to recall its desired capabilities. It must be able to both handle discrete and continuous variables, and deal with complex and thus potentially non-convex design spaces. Finally, it must be independent of predefined settings since the behavior of the response is completely unknown at this point. Table 7 compares and evaluates the aforementioned methods in order to find the most suitable algorithm. For the stated problem, the best technique appears to be NSGA-II.

While NSGA-II can be used to individually optimize each architecture, a process must be developed to integrate them into a single optimization process and consequently enable both optimization and comparison. This integration is discussed in the next section.

#### *2.1.3.4 Proposed Optimization Process*

This section discusses the development of an optimization process that combines all architectures into a single problem. As a first step, the previously mentioned architecture optimization problems are considered as sub-problems. Hence, the primary optimizer takes architectures as variables and outputs the overall Pareto frontier. The latter is generated by combining all sub-Pareto frontiers. The first idea would be to sequentially define the

**Table 7:** Comparison of the various multi-objective optimization algorithms

| | Discrete variables | Non-convex space | Pareto frontier | Best solution | No pre-defined settings |
|---|---|---|---|---|---|
| **OEC** | | ✓ | | ✓ | ✓ |
| **Weighted p-norm** | | | ✓ | | ✓ |
| **Epsilon-Constraint** | | | ✓ | | |
| **NBI** | | | ✓ | | ✓ |
| **NNC** | | | ✓ | | ✓ |
| **WSA** | ✓ | | ✓ | | ✓ |
| **VEGA** | ✓ | | ✓ | | ✓ |
| **NPGA** | ✓ | ✓ | ✓ | | |
| **SPEA** | ✓ | | ✓ | | ✓ |
| **NSGA** | ✓ | ✓ | ✓ | | |
| **NSGA-II** | ✓ | ✓ | ✓ | | ✓ |
| **MOGA** | ✓ | ✓ | ✓ | | |
| **PPES** | ✓ | ✓ | ✓ | | |
| **TGA** | ✓ | ✓ | ✓ | | |

Pareto frontier of each architecture and combine them. However, both alternative evaluation and NSGA-II are expensive to run. In addition, variables are all categorical since they correspond to the various architectures. To address these challenges, a new evolutionary algorithm inspired by simulated annealing is developed, that is able to handle multiple architectures. Simulated annealing is an optimization algorithm inspired by the annealing process for metals [123]. This type of metaheuristic algorithm enables an efficient search in presence of local minima. Besides, it enables a better efficiency than a simple grid search while keeping diversity and allowing uphill searches. Hence, the new evolutionary algorithm is defined by the following four-step process:

1. Start to execute the NSGA-II for each architecture with an initial number of generations $N_0$ identical for all architectures.

2. Evaluate the fitness $f_i$ of each architecture. $f_i$ corresponds to the number of non-dominated points originating from architecture $i$ in the overall Pareto frontier.

3. Determine the number of generations $N_i$ of each architecture for the next iteration of the primary algorithm based on the following principle:

   - If $f_i \neq 0$: $N_i$ is proportional to $f_i$.
   - If $f_i = 0$: $N_i$ has a probability $p$ to reach $N_0$. $p_i$ is defined in Equation 7, where $m_i$ is the number of fruitless iterations of architecture $i$ and $T$ a given constant.

$$p_i = 1 - \exp\left(\frac{-m_i}{T}\right) \tag{7}$$

4. Repeat steps 2 and 3 until the convergence criterion is met.

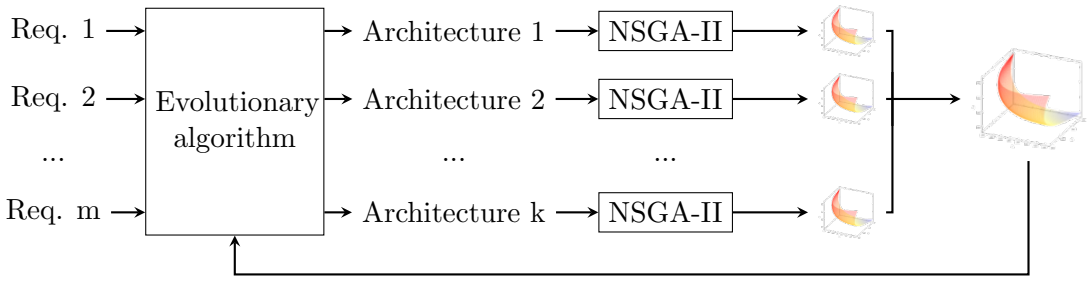



**Figure 20:** Description of EMMA

Implementing this algorithm has numerous advantages. It enables promising architectures to be favored in the selection process by assigning them more generations. Moreover, it also enables less promising architectures to be explored in order to ensure that every point of the design space has a chance to be explored independently of the initial conditions. Hence, all promising configurations can be investigated. The constant $T$ needs to be determined and represents the trade-off between fast convergence and insensitivity to the initial points, and accuracy of the final set of points. The overall optimization process, called Evolutionary Multi-architecture Multi-objective Algorithm (EMMA), is summarized in Figure 20 and leads to the formulation of the following Hypothesis:

**HYPOTHESIS 1.2: IF an evolutionary multi-architecture multi-objective algorithm based on architecture fitness is developed to drive a sequential use of multiple NSGA-IIs THEN the Pareto frontier of solutions defined by different sets of continuous, discrete, and categorical variables can be efficiently generated.**

In order to validate this Hypothesis, Experiment 1.2 is implemented. First, a list of all requirements as well as the output of Experiment 1.1 are needed. This experiment also requires the NSGA-II from Matlab and a design framework to evaluate each alternative. Next, the aforementioned design process will be implemented to find the overall Pareto frontier under fixed requirements. In parallel, a DoE will also be developed across all architectures to be used as a baseline for comparison. The validation of Hypothesis 1.2 requires three elements to be checked: the accuracy of the location of the Pareto frontier, the quality of its sampling, and its usability for complex and large-scale problems. This will be done through three questions:

1. Are most of the points from the DoE dominated by the points on the Pareto frontier of the proposed algorithm?

2. Is the Pareto frontier evenly sampled?

3. Does the benefits of the proposed algorithm increase with the problem complexity?

#### *2.1.3.5 Summary of the New Design Space Exploration Method*

The combination of the two methodologies developed for both alternative generation and alternative comparison/optimization are now combined to formulate the following Hypothesis, which addresses Research Question 1.

**HYPOTHESIS 1: IF all feasible alternatives are systematically generated using a variable-oriented morphological analysis AND IF they are simultaneously compared and optimized using an evolutionary multi-architecture multi-objective algorithm based on architecture fitness THEN large design spaces can be better explored at a conceptual design level.**

In order to validate this Hypothesis, one must ensure that the combination of the two methods described in Experiments 1.1 and 1.2 fully addresses Research Question 1. Experiment 1 will ensure that the design space exploration has been improved by comparing its results to existing methodologies. In particular, the proposed methodology must enable

new promising concepts to be rapidly identified in a design space, where alternatives are not necessarily defined by the same variables and/or evaluated with the same modeling and simulation environment.

Hitherto, requirements were considered as fixed. However, as discussed in Section 1.3, they are highly uncertain and their uncertainty evolves over time. This evolving uncertainty will be discussed in the following section.

## *2.2 Decision-Making Under Evolving Uncertainty in Requirements*

Accurate requirements definition is one of the key success factors of a good design. However, it is also one the most challenging parts of the design process. This task becomes even more difficult with the presence of the various sources of uncertainty described in Section 1.3. Traditional aerospace design processes often rely on a single design mission [45] that best represents the most constraining mission requirements. However, new vehicles have to be designed while requirements (mission, safety, ticket price, and comfort) have not been defined yet. In addition, such vehicles could support various future applications. Raymer defines aerospace design as the process of finding the vehicle that meets or maximizes all other requirements when sized to the design mission [354]. This would result in having one vehicle per mission, which is not economically viable. The concept of robust design has therefore been introduced and refers to techniques that attempt to mitigate the effects of variability of different factors on the vehicle performance. Section 1.3 also established the evolving nature of this uncertainty. Indeed, at the beginning of a market establishment, uncertainty in requirements is extremely high and there is no most probable value. Along with technology maturation and market establishment, various possible applications appear and thus provide different goals for requirements and regulations. These goals are translated into most probable value around which uncertainty is still strong. Finally, at the dawn of the detailed design and the production phases, potential applications are frozen and regulations are precise enough. Figure 21 illustrates this evolution in regulations and applications along with its impacts on uncertainty. In this example, the first requirement

converges towards a single value while the second requirement converges towards two different end values due to different applications.

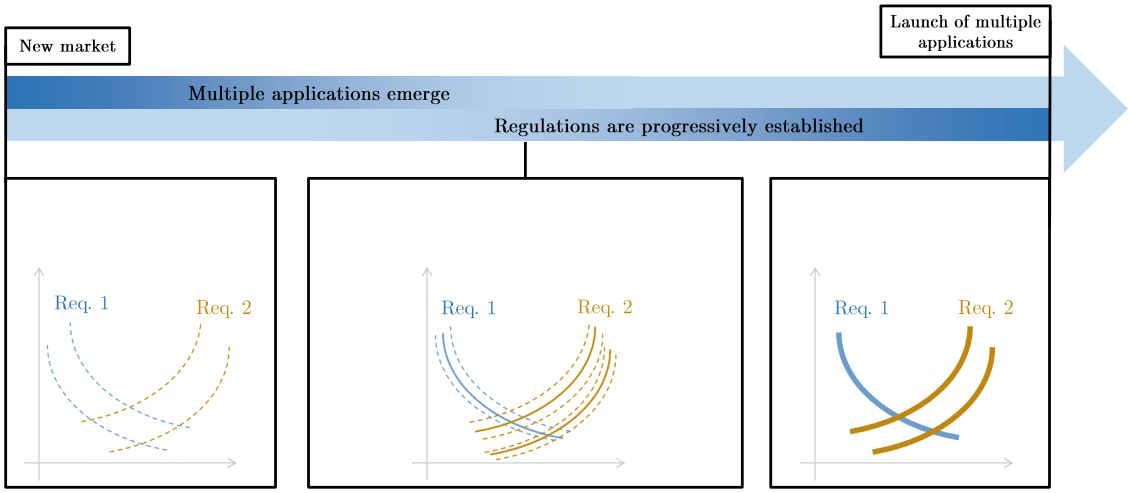



**Figure 21:** Evolution of requirements' uncertainty over time

The following sections describe the steps of a decision-making methodology that captures all periods of the market establishment.

### 2.2.1 Gap Identification

Based on the previous analysis, three different states can be identified and are represented in Figure 22. While States 1 and 3 are static, State 2 is dynamic since both the degree of uncertainty and the most probable value change.

(a) State 1 (b) State 2 (c) State 3

**Figure 22:** Different states of requirements

Therefore, to address this second state, designers must be able to rapidly control both the degree of uncertainty and the most probable values. To be taken into account, uncertainty must be propagated through the design framework and its consequences must be assessed. Due to its complexity, the design framework has a long execution time so that the treatment of uncertainty must rely on few function calls. Finally, since uncertainty distributions are usually unknown at the beginning of the market development, the process should not rely on predefined distributions to propagate uncertainty. To discuss the way uncertainty can be modeled and propagated, this section describes, compares, and evaluates state-of-the art techniques in robust design. These methods can be decomposed in five categories: probabilistic methods, non-probabilistic methods, dynamic set-based design methods, information-gap decision theory, and Bayesian Information Toolkit (BIT).

#### *2.2.1.1 Probabilistic Approach*

The idea of probabilistic approaches is to propagate uncertainty by treating noise variables as random variables modeled by probability distributions instead of fixed values [172]. Figure 23 describes the general process of probabilistic approaches.

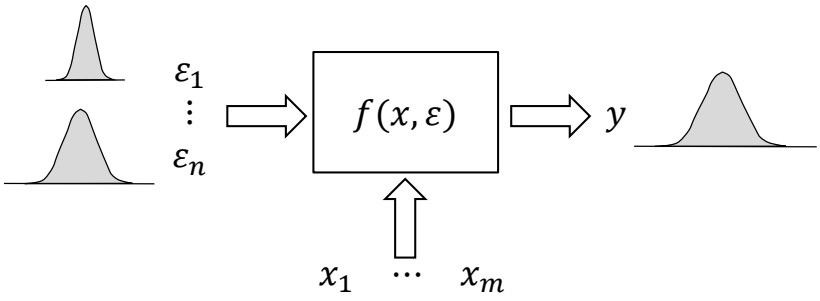



**Figure 23:** General process of probabilistic robust design [170]

The implementation of probabilistic robust design can be decomposed into three different steps: noise variable modeling, uncertainty propagation, and robust design formulation. Each of these steps requires the choice of a method to be implemented.

**Noise variable modeling:** As previously mentioned, variables must be modeled by probability distributions. Depending on the circumstances, one of the following distributions may be used [285]:

- Uniform: for a given range $[l; u]$, all outcomes $x$ are equally likely. This represents a complete uncertainty and it can be modeled using Equation 8.

$$u(x) = \begin{cases} \frac{1}{u-l} & \forall\, l \leq x \leq u \\ 0 & \text{elsewhere} \end{cases} \tag{8}$$

- Triangular: used when three outcomes are known. In particular, a triangular distribution is characterized by the minimum $l$, the maximum $u$, and the most likely $m$. It can be modeled using Equation 9.

$$t(x) = \begin{cases} \frac{2(x-l)}{(u-l)(m-l)} & \forall\, l \leq x \leq m \\ \frac{2(u-x)}{(u-l)(u-m)} & \forall\, m \leq x \leq u \\ 0 & \text{elsewhere} \end{cases} \tag{9}$$

- Gaussian: represents the addition of many random numbers and is widely used to represent natural phenomena. It is modeled in Equation 10, where $\mu$ is the mean and $\sigma^2$ the variance.

$$g(x) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp^{-\frac{1}{2\sigma^2}(x-\mu)^2} \tag{10}$$

- Weibull: usually used to model reliability and failure rates in engineering. It is modeled in Equation 11, where $\alpha$ and $\beta$ are two constants so that the failure rate is expressed as $Z(t) = \alpha\beta t^{\beta-1}$.

$$w(x) = \begin{cases} \alpha\beta x^{\beta-1} \exp^{-\alpha x^{\beta}} & \forall\, x > 0 \\ 0 & \text{elsewhere} \end{cases} \tag{11}$$

- Beta: one of the most flexible distributions used to model random variables within a given range $[l; u]$. It can be modeled using Equation 12, where $\alpha$ and $\beta$ are two fixed integers.

$$b(x) = \frac{(u-x)^{\beta-1}\,(x-l)^{\alpha-1}}{\frac{(\alpha-1)!(\beta-1)!}{(\alpha+\beta-1)!}\,(u-l)^{\alpha+\beta-1}} \tag{12}$$

Once one of these distributions has been selected for each noise variable, they need to be propagated through the optimization process. The different methods are discussed below.

**Uncertainty propagation:** Uncertainty propagation refers to the way used to assess the effect of the variability of each noise variable on the objective function(s). Two types of methods can be identified: statistical and non-statistical methods [84].

Statistical methods use a large number of samples of the noise variables and repeatedly analyze the output of the system to statistically determine its distribution. Among statistical methods, the most widely used are the Monte-Carlo simulation and the Latin Hypercube sampling method. The Monte-Carlo simulation generates points by randomly sampling the noise variables based on their probability distribution. Then, it executes the code to find the output corresponding to each sample. Finally, it statistically creates the Probability Density Function (PDF) of the objective function. Even though this method is very accurate as it provides the entire PDF, it is computationally expensive because the code must be executed for a large number of points. This technique is widely used within the aerospace domain [160, 272, 275, 359]. The Latin Hypercube sampling method wisely generates sample points based on a division of each variable into intervals [288]. While the number of points of the final sampling can be chosen by the user, each interval can only be represented once in the final sampling. Even though it is more computationally efficient than the Monte-Carlo simulation, it still requires many function calls. Hence, to address this common pitfall of statistical methods, non-statistical methods have been developed.

While being more efficient, non-statistical methods are also less accurate and require several assumptions to be made. Examples of such methods are Moment methods, polynomial chaos, and Bayesian Monte-Carlo (BMC) technique. Moment methods use a Taylor series expansion to approximate the response variance as a function of the variances of each input parameter. Equations 13 and 14 present the First-Order Second Moment (FOSM) method and the Second-Order Second Moment (SOSM) method, respectively.

$$\mathrm{Var}_1\left(y(x)\right) = \mathrm{Var}(x)\left(\left.\frac{\partial y}{\partial x}\right|_{\bar{x}}\right)^2 \tag{13}$$

$$\mathrm{Var}_2\left(y(x)\right) = \mathrm{Var}(x)\left(\left.\frac{\partial y}{\partial x}\right|_{\bar{x}}\right)^2 + \frac{1}{2}\left(\mathrm{Var}(x)\left.\frac{\partial^2 y}{\partial^2 x}\right|_{\bar{x}}\right)^2 \tag{14}$$

While these methods are computationally efficient and thus useful for design space exploration, they have some pitfalls. Indeed, they highly depend on the type of PDF chosen for the noise variables and are only accurate for Gaussian distributions. To address the second drawback, polynomial chaos expansions decompose the uncertainty into deterministic and random components. The random component is characterized by an infinite series of Hermite polynomials. BMC incorporates prior knowledge into the process using Gaussian process based on Bayes Theorem and consequently improves the classical Monte-Carlo simulation. Equation 15 presents this theorem for a single uncertain parameter $\theta$ and the observed parameter $x$.

$$P\left(\theta|x\right) = \frac{P\left(x|\theta\right)P\left(\theta\right)}{P\left(x\right)} \tag{15}$$

While this method is also relatively efficient, it still requires prior knowledge about the distribution of the noise variables and many function calls. This method is also more complicated to implement and depends on some predefined parameters that could impact the accuracy of the final result. Once the probability distributions have been propagated through the design process, the final step consists in defining an objective function that incorporates the variability.

**Objective function formulation:** The objective function aims at balancing the optimization of the initial objective function and the minimization of the variability. Three main approaches are commonly used: traditional robust design formulation, robust design Pareto frontiers, and reliability-based design optimization [172]. Traditional robust design formulations define a new objective function presented in Equation 16 that needs to be minimized. In this equation, $\mu_y$ is the mean of the objective function, $y_T$ its targeted value, and $\sigma_y^2$ its variance.

$$\min\left(\mu_y - y_T\right)^2 + \sigma_y^2 \tag{16}$$

This formulation can also be extended to multi-objective optimization using the Joint Probabilistic Decision Making (JPDM) [27, 28]. The latter introduces the probability of success

as a new objective function that contains information from all other objective functions. Robust design Pareto frontier techniques aim at determining the set of non-dominated solutions with respect to both the objective function and its variance. Finally, reliability-based design optimization focuses on guaranteeing that a given constraint $g_i$ is not violated within a specific probability $R_i$. Its general formulation is given in Equation 17, where $x$ are the design variables and $\epsilon$ the noise variables.

$$P\left[g_i(x,\epsilon)\right] \leq R_i \tag{17}$$

While various methodologies have been presented to model, propagate, and analyze uncertainty, they all rely on user-defined probability distributions. However, these distributions are usually unknown at the beginning of the establishment of a new program. The next section presents other robust design methodologies that do not require uncertainty distributions.

#### *2.2.1.2 Non-Probabilistic Approach*

Non-probabilistic methods allow decision makers to consider uncertainty variables without precise information about their probability distributions. Two main methods exist in the literature: fuzzy set theory and interval analysis.

**Fuzzy set theory:** Fuzzy set theory assesses the behavior of a system under inexact or unreliable variables [80, 125, 126]. This methodology is especially useful when variables cannot be modeled statistically. In addition, it is very simple to use and to implement. Hence, it has been widely used for engineering applications.

Fuzzy logic calculates the approximate behavior of the system using models based on inexact or unreliable data. The membership function represents the degree of membership of the fuzzy variable within a fuzzy set. First introduced by Zadeh in 1965, membership functions assign to each object a grade of membership between zero and one [491]. They enable the modeling of imprecision in definition criteria and consequently represent the degree of truth. Uncertainty analysis using fuzzy sets is often called a possibility approach.

Fuzzy logic has been widely used in engineering applications [36, 124, 137, 245, 267, 397, 495, 496, 497].

To illustrate how fuzzy set theory propagates uncertainty, one can consider a problem with $n$ uncertain design variables $x_i$. These variables are modeled by membership functions, which represent how much a variable is in a set. Probability represents how probable it is that a variable is in a set. Hence, in fuzzy set theory, axioms of classical probability can be relaxed. Moreover, each variable is characterized by a mean value $\mu_i$ and a standard deviation $\sigma_i^2$. If $X$ is defined as the objective function such as $X = f(x_1, x_2, ...x_n)$, the robust design problem can be defined using Equation 18. In this equation, $T$ is the target value of the objective function, $g_i = \frac{\partial f}{\partial x_i}$, $h_{ij} = \frac{\partial^2 f}{\partial x_i \partial x_j}$, and $\sigma_{ij}$ is the covariance of $x_i$ and $x_j$.

$$\begin{aligned} \min \mathrm{Var}(X) &= \min \sum_{i=1}^{n} \sum_{j=1}^{n} g_i(\mu) g_j(\mu) \sigma_{ij} \\ \text{subject to } \ \mathrm{E}(X) - T &= f(\mu) + \tfrac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} h_{ij}(\mu) \sigma_{ij} - T \end{aligned} \tag{18}$$

A consequence of this simplicity is a reduced accuracy compared to probabilistic methods. Indeed, instead of propagating the entire distribution, only the mean and the variance are propagated. However, for a first order estimate, this methodology appears ideal since both the mean value and the degree of uncertainty can be controlled.

**Interval analysis:** Interval analysis models uncertainty using two parameters: the minimum and maximum values constraints are expected to take [113, 118, 119]. Hence, if $P = [p_1, p_2, ..., p_t]$ is the vector of constraints, its uncertainty $P^I$ can be defined with $P^c$ and $P^w$, which are the middle vector and the radius vector, respectively, as described in Equation 19.

$$P^I = P^c + [-1\,,\, +1]\, P^w \tag{19}$$

Based on this definition, the initial optimization problem is translated into a bi-objective optimization problem as illustrated in Equation 20. For a specific optimal solution $X$, $F^L$ and $F^U$ represent the minimum and the maximum of the interval number $F\left(X, P^I\right) =$

$f(X,P)$, respectively.

$$\begin{aligned} \min f(X,P) &= \min\left(F^c, F^w\right) \\ F^c &= \tfrac{1}{2}\left[F^L\left(X,P^I\right) + F^U\left(X,P^I\right)\right] \\ F^w &= \tfrac{1}{2}\left[F^U\left(X,P^I\right) - F^L\left(X,P^I\right)\right] \end{aligned} \tag{20}$$

This method, easy to implement, only considers the bounds of the uncertainty interval and does not allow decision makers to favor a specific most probable value. In addition, it has been shown that, for large problems, the solution might be too conservative [84].

#### *2.2.1.3 Dynamically Constrained Set-Based Design*

Set-based design has been developed by Toyota and improves traditional point-based design [405]. It is based on three main principles. First, it explores the design space by defining the feasible region for each discipline. Then, it combines all disciplines to determine the feasible design space of the vehicle, which is the intersection of all previous regions. Several constraints can also be added to narrow this design space. Finally, it gradually reduces disciplinary feasibility regions by increasing the level of detail so that the feasibility region is defined before commitment. Kizer et al. improved this methodology to integrate uncertainty in requirements/constraints and thus developed the Dynamically Constrained Set-Based Design [238]. Traditional set-based design has been adapted assuming non-static requirements. A three-step method is described:

1. Creation of the feasible design space similarly to conventional set-based design space using nominal values for the constraints.

2. Perturbation of the constraints to obtain an intermediate approximation of the robust feasible design space.

3. Refinement of the robust design space.

This methodology allows designers to identify robust designs and develop scenarios with different constraints. Hence, it enables the identification of both critical constraints and main design drivers. However, it does not enable the simultaneous control of both the most probable value and the degree of uncertainty. This approach can be used to address States 1 and 3 of the requirements but not State 2.

#### *2.2.1.4 Information-Gap Decision Theory*

Information-gap decision theory aims at supporting decision makers in the presence of strong uncertainty. This non-probabilistic method represents the disparity between what is known and what is unknown. The theory is based around three models: an uncertainty model, a system model, and a decision-making model.

**Uncertainty model:** This model quantifies uncertainty in a non-probabilistic way [32, 83, 203]. To do so, it uses $\alpha$, called horizon of uncertainty, in order to model the nested subsets $U(\alpha, \tilde{u})$ around the estimate of the parameter $\tilde{u}$. The size of this subset increases with uncertainty and $\alpha$ measures the distance between the estimate and a possibility. This theory can be considered as a local decision theory since it starts with an estimate and then applies deviations around it. For instance, if a variable $u$ is expected to take the value $\tilde{u} = 100$ but this information is only reliable at $\pm 10\%$, the nested subset is given in Equation 21, where $\alpha = 10$.

$$U(\alpha, \tilde{u}) = \{u :| u - \tilde{u} |\leq \alpha\} \tag{21}$$

**System model:** This model enables uncertainty to be propagated through the system by modeling the behavior of the real system. Since it represents designers' theoretical knowledge, it might also include uncertain elements. It corresponds to the design framework or the performance analysis code. Then, for a given minimum/maximum level of the desired outcome, the system will determine the maximum degree of uncertainty acceptable to reach this value. This corresponds to the robustness of the decision and assesses the greatest level of uncertainty under which requirements are always met. It represents the ability of the system to achieve acceptable outcomes over a large range of noise factor inputs. In addition, for a given windfall output, the system will determine the level of uncertainty required to reach this value. This corresponds to the opportuneness of the decision and forecasts the level of uncertainty that ensures success.

**Decision-making model:** This model specifies the targeted values of outcomes. It ensures the success of the decision by deciding the trade-off between the two aforementioned objectives. The robustness optimization implies the maximization of the level of uncertainty under which the required outcome would fail. The opportuneness optimization requires the minimization of the horizon of uncertainty constrained to a given value of the objective function.

The advantage of this theory is its ability to predict uncertainty without using probability distributions. However, this theory cannot capture unexpected events since it is based on a fixed universe of possibilities. In addition, it requires a first estimate as a starting point, which is not necessarily available in early stages. Hence, at this stage, it might be preferable to consider the entire region of uncertainty rather than deviations around an estimate. Finally, because it is a static process, it does not enable trade-offs and does not match the dynamic behavior of decision-making [402].

#### *2.2.1.5 Bayesian Information Toolkit*

While all previous methodologies consider uncertainty analysis as a static approach, BIT aims at developing a more flexible ability to adapt the design when new data sets become available [469]. This method combines probability encoding methods and Bayesian updating techniques. Three main probability encoding methods are suggested to systematically extract information from subjective information and experts' judgment to uncertain quantities. The variations depend on what is asked: probability (P), values (V) or both (PV):

- P-method: it requires specific points on probability scale for fixed values.
- V-method: it requires specific values for fixed probability.
- PV-method: it requires the description of points on the cumulative probability distribution.

Bayesian updating techniques are based on Bayes' theorem. Let $X$ be a random variable and $\theta$ a realization of the random variable $\Theta$. The probability density function of $X$ is

$f(x,\theta)$ and the one of $\Theta$ is $f_\Theta(\theta)$. To update information about $\Theta$ using observation about $X$, the Bayes' theorem is expressed using Equation 22.

$$f_{\Theta|X}(\theta|x) = \frac{f_{X,\Theta}(x,\theta)}{f_X(x)} = \frac{f_{X|\Theta}(x|\theta)\, f_\Theta(\theta)}{f_X(x)} \tag{22}$$

Then, a six-step methodology is provided to propagate uncertainty that integrates the two previous methods [469]:

1. Identify variables subject to uncertainty.
2. Select probability encoding methods.
3. Collect data.
4. Choose Bayesian updating models.
5. Translate subjective data into model parameters by inverse Cumulative Distribution Function (CDF) analysis.
6. Update uncertainty model.

While this methodology enables both mean values and uncertainty to be controlled, several pitfalls have been identified. Indeed, this approach is based on probability distribution functions, which are usually unknown in early phases. In addition, it only provides a way to update the model but does not allow to rapidly develop multiple scenarios. Finally, uncertainty is propagated using statistical methods and consequently the execution time will be too long if each function evaluation already takes a large amount of time.

This review of current methodologies is summarized in Table 8, which assesses the performance of each methodology with respect to important criteria: the ability to control both the mean value and the degree of uncertainty, the ability to evaluate robustness without requiring probability distributions, and the computational efficiency of the process. Table 8 shows that there is no available method that fully meets all required capabilities. Indeed, most of the existing methodologies are static and can only address either State 1 or State 3. The only methodology that is flexible (Bayesian Information Toolkit) requires to input

probability distributions and consequently requires knowledge about uncertainty. Moreover, this method is long to run and does not match the potential long execution time of the evaluation environment.

**Table 8:** Comparison of the various robust design methodologies

| | **Controllable degree of uncertainty** | **Controllable values** | **Distribution not required** | **Fast** |
|---|---|---|---|---|
| **Conventional probabilistic methods** | ✓✓ | | | |
| **Conventional non-probabilistic methods** | ✓✓ | | ✓✓ | ✓✓ |
| **Dynamic SBD** | | ✓✓ | ✓✓ | ✓ |
| **Info-gap theory** | ✓✓ | | | |
| **BIT** | ✓ | ✓ | | |

This leads to the formulation of the following Research Question:

**RESEARCH QUESTION 2: How can decision makers identify and prioritize a set of solutions robust to evolving uncertainty in requirements?**

This Research Question can naturally be decomposed into two distinct goals that will sequentially be discussed in the next two sections. First, evolving requirements' uncertainty must be addressed and then objective prioritization must be enabled.

### 2.2.2 Modeling of Evolving Requirements' Uncertainty

The two elements that might evolve across the three states are the degree of uncertainty and the mean values of requirements. The idea of the new process proposed in this research is to sequentially address each capability.

In order to model and propagate uncertainty around fixed values, either conventional probabilistic or non-probabilistic methods can be used. To find the most suitable one for

this problem, different methods are compared in Table 9 against the required capabilities: number of function evaluations (speed), information required about the uncertainty (entire distribution or few data), accuracy of the result, their ability to use the deterministic nature of the problem (do not repeat cases), and their usability for large problems.

**Table 9:** Comparison of the various uncertainty propagation methods

| | **Fast** | **No distribution required** | **Accuracy** | **Use deterministic nature** | **Usable for large problems** |
|---|---|---|---|---|---|
| **Monte-Carlo** | | | ✓✓ | | ✓✓ |
| **Latin Hypercube** | | | ✓✓ | | ✓✓ |
| **Baysian Monte-Carlo** | | | ✓✓ | ✓ | ✓✓ |
| **FOSM & SOSM** | | | ✓✓ | ✓ | ✓✓ |
| **Interval analysis** | ✓✓ | ✓✓ | ✓ | ✓✓ | |
| **Fuzzy logic** | ✓✓ | ✓✓ | ✓ | ✓✓ | ✓✓ |

Based on this comparison, fuzzy logic seems to be the most suitable approach to model and propagate uncertainty around fixed values. To modify these fixed values, a DoE can be used as already implemented in Dynamically Constrained Set-Based Design. Being more efficient than a full factorial, DoEs are smarter methods to cover the design space. Hence, once combined, decision makers can vary both the mean value of each requirement to develop multiple scenarios and the degree of uncertainty of the membership function through the variance of this membership function using the DoE. If this methodology is combined with a design space exploration, new trends can be identified and new capabilities are available to help designers make decisions under evolving uncertainty. The next section aims at incorporating those two methodologies into a framework that also allows designers to prioritize their objectives and better support their decisions.

### 2.2.3 Objective Prioritization

Objective prioritization is performed in a discrete design space since both the sampling of the Pareto frontier and the DoE provide discrete values. Hence, Multi-Attribute Decision Making (MADM) techniques can be used. The objective is to rank the different alternatives according to stated criteria to support decision makers. There are four common steps to all MADM techniques that must be carefully followed to ensure an efficient process [330]:

1. Select the relevant criteria and alternatives.
2. Quantify the relative importance of each criterion.
3. Evaluate each alternative with respect to each criterion.
4. Determine a ranking of alternatives.

While a large number of techniques exist, they have been categorized with respect to the type of information they can handle: no information, information on environment, and information on attributes. Since the first two types are often too subjective, the last one is preferred in this research. This category of techniques can be further decomposed according to the characteristics of the information itself: ordinal, cardinal or nominal. For optimization purposes and accuracy, methods based on cardinal data are preferred. Indeed, if no cardinal data is used, a concept which is slightly better than the others for all criteria except for one (for which it is drastically worse) could be considered as the best one. Hence, only MADM techniques based on cardinal data are reviewed and compared.

#### *2.2.3.1 Pugh Decision Matrix*

Created by Stuart Pugh, the decision-matrix method is a method that quantitatively compares multiple design alternatives [349]. It leads to the best alternative with respect to an established set of design criteria. The Pugh Matrix is one of the most popular examples of such decision matrices [62, 70, 210, 404, 438]. Once these criteria have been identified, a baseline is set. Then, each alternative is benchmarked against this datum point using three different symbols: "+" (better than the baseline), "-" (worse than the baseline), "S" (same

as the baseline). To grade each concept, "+1" is assigned to "+", "-1" to "-", and "0" to "S". Then, individual concept scores are calculated and the best concept is selected. A notional Pugh Matrix is presented in Table 10. According to this notional example, Concept 1 should be selected among the ones evaluated.

**Table 10:** Notional Pugh Matrix

| Alternatives / Criteria | **Concept 1** | **Concept 2** | **Concept 3** | **Baseline** |
|---|---|---|---|---|
| **Maximum altitude** | + | + | S | |
| **Micro-gravity time** | + | - | - | |
| **Maximum load factor** | S | + | - | |
| **Ticket price** | + | - | S | |
| **Emissions** | S | S | + | |
| **Total score** | **+3** | **0** | **-1** | |

This technique is able to handle a large number of decision criteria and concepts in a very simple manner. Nevertheless, an unwise choice of the baseline (worse/better than a majority of the concepts) would imply a lack of strong pattern. In this situation, the datum point must be changed and the process restarted. Also, it does not take into account quantitative data even if they are available. For example, a concept with a huge ticket price (ten times higher than the others) could appear to be the best if all other criteria are slightly better than the baseline. Finally, it does not allow decision makers to give more/less importance to some specific criteria.

#### *2.2.3.2 Technique for Order Preference by Similarity to Ideal Solution*

The Technique for Order of Preference by Similarity to Ideal Solution (TOPSIS) is based on the assumption that the best alternative has the shortest distance to the utopia solution [247, 282]. This approach first defines the utopia and the anti-utopia solutions, which are respectively a combination of the best and the worst alternatives for each criterion. Then, the Euclidean distance between each concept and those two points is calculated. The ranking of all alternatives is consequently obtained: from the best (closest to the utopia point) to the worst (farthest to the utopia point). The distance is computed using a utility

function as presented in Equation 23. The weighting factor $w_i$ represents the importance of the i$^{\text{th}}$ criterion and $y_i^j$ the value for this criterion for the j$^{\text{th}}$ concept. This value is usually normalized in order to ensure consistent results while comparing small and large numbers such as take-off gross weight and sweep angle. Moreover, if no cardinal information exists, a scale can be created to quantify all objective values. For example, one can evaluate safety on a scale from 0 to 10, 0 being the worst and 10 the best.

$$U(y^j) = \sum_i y_i^j w_i \tag{23}$$

While providing the advantages of the Pugh Matrix (simplicity and ability to compare a large number of criteria/concepts), the TOPSIS makes up for some of its drawbacks. Indeed, the weighted sum allows decision makers to define their preferences on specific aspects (cost, performance, etc.). Besides, it uses absolute values rather than a relative comparison with the baseline. However, this technique assumes a monotonic behavior of the evaluation criteria and requires subjective and fixed values for the different weighting factors $w_i$. Modifying those weighting factors can be analogous to stretching the corresponding axes. It might become problematic and lead to biased solutions as the optimum solution is often very sensitive to these weights.

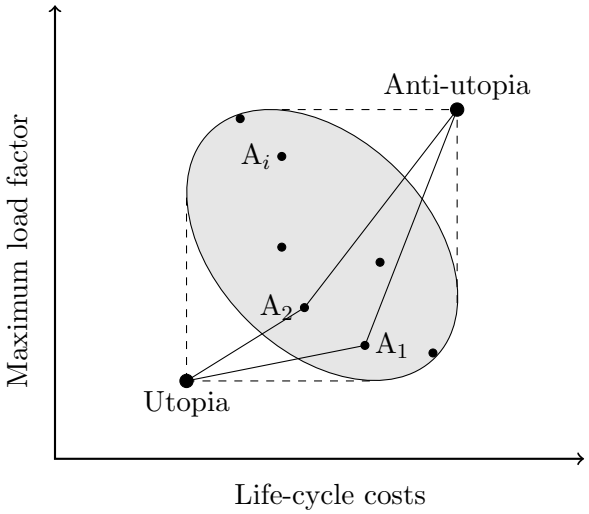



**Figure 24:** Illustration of a two-objective TOPSIS application

Figure 24 shows a two dimensional example of the TOPSIS application. $A_i$ are the different alternatives that define the design space drawn in blue. The goal is to minimize the two objectives (costs and load factor) but there is a trade-off: the alternative with the minimum load factor does not correspond to the cheapest alternative.

#### *2.2.3.3 Elimination and Choice Translating Reality*

The "Elimination and Choice Translating Reality" (ELECTRE) technique, which originally comes from the French "Elimination Et Choix Traduisant la Réalité" is a method that dichotomizes alternatives based on outranking relationships. It is based on binary comparisons between two alternatives and aims at determining which alternative is better (outranks) than the other even if there is no strict domination. Domination means that an alternative is better than another in all criteria. It can be noted that this relationship is not transitive: even if $A_1$ outranks $A_2$ ($A_1 \rightarrow A_2$) and $A_2$ outranks $A_3$ ($A_2 \rightarrow A_3$), there is no indication that $A_1$ outranks $A_3$ ($A_1 \rightarrow A_3$). To understand how preferred alternatives are chosen, an example with eight alternatives $A_i$ is taken and a diagraph (Figure 25) is built according to the outranking relationship principles [485].

Once this diagraph has been drawn, ELECTRE seeks the kernel $K$, which is the core of the diagraph. Every node that belongs to this kernel is a preferred alternative. $K$ has two properties: it does not contain any node that is outranked by another one and every node that does not belong to the kernel must be outranked by at least one other node in $K$. In the case illustrated in Figure 25, the kernel is defined by the nodes $A_1$, $A_2$ and $A_5$. Finally, by developing concordance and discordance indexes, ELECTRE quantifies the outranking relations and enables the automation of this methodology. One of the main drawbacks is the lack of ranking among the kernel's alternatives and consequently there is no indication about "the best" solution. Indeed, all alternatives within the kernel are considered to be at the same level. Moreover, in order to formalize outranking relationships, thresholds are set and their arbitrary definition highly impacts the final results.

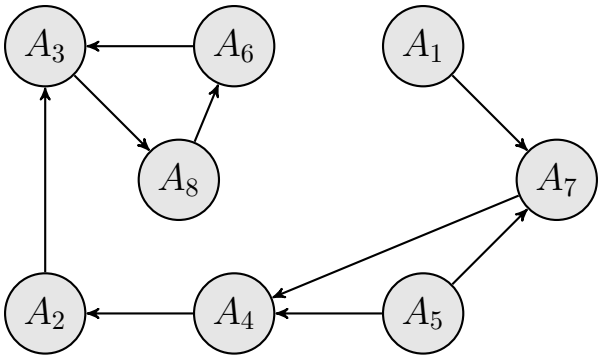



**Figure 25:** Diagraph of the eight alternatives [485]

#### *2.2.3.4 Analytical Hierarchy Process*

The Analytical Hierarchy Process (AHP) tries to structure expert judgments to make rational decisions [152]. This approach only requires comparisons between alternatives' attributes. Its applications are multiple and include choices, ranking, prioritization, resource allocation, etc. The process can be decomposed into five steps:

1. Define the $n$ criteria and the $m$ alternatives of interest. This includes gathering information about each alternative and defining a scale from 1 to 9 to evaluate alternatives with respect to each criterion.

2. Establish priorities among criteria by assigning weighting factors $w_i$ to each of them.

3. Create a comparison matrix $M_i$ for each criterion as presented in Equation 24. Each element at the j$^{\text{th}}$ row and k$^{\text{th}}$ column of this square matrix corresponds to the ratio of the score of the j$^{\text{th}}$ alternative to the score of the k$^{\text{th}}$ alternative.

$$M_i = \begin{pmatrix} 1 & \frac{A_{i,1}}{A_{i,2}} & \cdots & \frac{A_{i,1}}{A_{i,m}} \\ \frac{A_{i,2}}{A_{i,1}} & 1 & \cdots & \frac{A_{i,2}}{A_{i,m}} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{A_{i,m}}{A_{i,1}} & \frac{A_{i,m}}{A_{i,2}} & \cdots & 1 \end{pmatrix} \tag{24}$$

4. Normalize each column of $M_i$ to create $\overline{M_i}$ and compute the priority vector $p_i$ by averaging each row.

5. Compute the final score for all alternatives in the final vector $f$ defined in Equation 25.

$$f = \sum_i w_i p_i \tag{25}$$

This method is powerful and flexible since it reduces highly complex decisions to simple one-on-one comparisons and can handle both qualitative and quantitative information. However, some inconsistencies could occur as each alternative is compared with all others. As a consequence, good scores for some criteria and bad scores for other criteria can be compensated due to aggregation and precious information can be lost. Moreover, the nine-point scale has limitations since it can be difficult for decision makers to decide whether an alternative is four or five times better than another. Finally, an alternative with an extremely bad score with respect to one criterion (that should eliminate the alternative) could still be promising due to the limitation of the nine-point scale.

According to this analysis, the TOPSIS seems to be the most suitable technique for this application. Indeed, this method is fast, easy-to-implement, and is capable of handling absolute values. Decision makers can also easily change their priorities in order to get the best solution. Now that all required pieces have been selected, the next section describes the complete methodology that addresses Research Question 2.

### 2.2.4 Proposed Decision-Making Process

The methodology developed for Research Question 2 is illustrated in Figure 26.

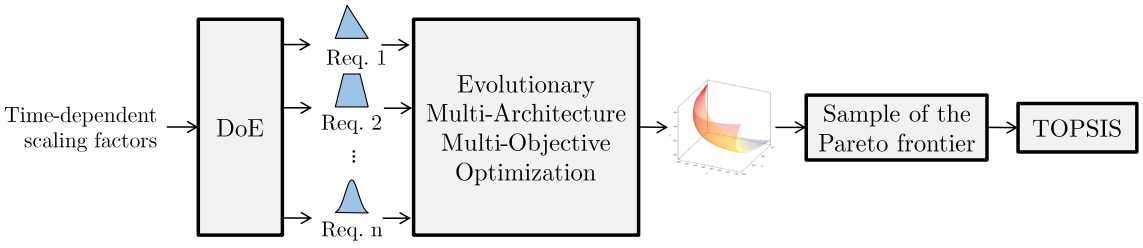



**Figure 26:** Overview of the proposed decision-making process

This process is built around the multi-objective optimization algorithm discussed in Section 2.1, which is treated as a black box responsible for providing the Pareto frontier of solutions under a given series of fixed requirements.

The decision-making methodology presented above can be decomposed into five steps, as detailed below:

1. Model uncertainty using membership functions.
2. Implement the algorithm for uncertainty propagation using fuzzy logic.
3. Create a time-dependent model for each membership function using scaling factors for the main parameters such as mean value and standard deviation.
4. Develop a DoE on the time-dependent scaling factors of each membership function.
5. Implement the TOPSIS based on the results given by the Pareto frontier and for each point of the DoE.

This process provides all the elements that help formulate Hypothesis 2.

**HYPOTHESIS 2: IF fuzzy set theory is used to propagate requirements' uncertainty whose magnitude has been modeled by scalable time-dependent membership functions AND IF a Design of Experiments of these scaling parameters is used to further create a TOPSIS THEN informed decisions can be made under fuzzy objectives and evolving uncertainty in requirements.**

In order to validate Hypothesis 2, Experiment 2 is implemented. As inputs, it requires a list of requirements modeled by time-dependent membership functions, objectives, an evaluation environment, and the process developed in Section 2.1 to find the Pareto frontier under specific requirements. The experiment consists in implementing the fuzzy set theory algorithm for uncertainty propagation, the DoE, and the TOPSIS. To fully validate Hypothesis 2, two elements need to be validated: the ability of the method to address evolving uncertainty in requirements and its ability to help designers with decision-making.

## *2.3 Formulation of the Overarching Hypothesis*

The capabilities developed through the new design space exploration process discussed in Section 2.1 and the process detailed in Section 2.2 to model and propagate evolving uncertainty in requirements are combined to formulate the Overarching Hypothesis presented below:

**OVERARCHING HYPOTHESIS: IF a variable-oriented morphological analysis is used to feed an evolutionary multi-objective multi-architecture optimization algorithm AND IF fuzzy set theory is used to parametrically propagate requirements' uncertainty through a multi-disciplinary physics-based modeling and simulation environment THEN large multi-architecture design spaces can be better explored AND informed decisions can be made under evolving uncertainty in requirements.**

In this chapter, several gaps have been identified and formulated through Research Questions. A literature review also highlighted some methods that have been leveraged to develop the cornerstones of the new methodology by the formulation of the two main Hypotheses and the Overarching Hypothesis. The following chapter organizes the aforementioned pieces into an approach that will enable the implementation of the Experiments in order to validate the corresponding Hypotheses and address the Research Objective.

# CHAPTER III

# PROPOSED APPROACH

This research focuses on establishing a methodology that enables a broad design space exploration at a conceptual level to select solutions against unclear objectives and under evolving uncertainty in requirements. Chapter 2 identified two main gaps in current practices that need to be bridged in order to meet the established needs. The research follows the generic top-down design decision support process presented in Figure 27.

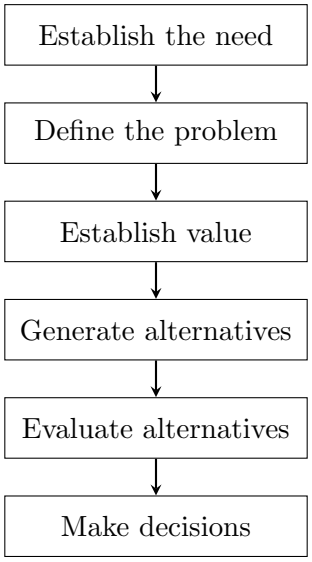



**Figure 27:** Generic top-down design decision support process

Chapters 1 and 2 aimed at addressing the first two steps of this decision process. Indeed, by looking at current practices and analyzing emerging markets, Chapter 1 led to the formulation of three assertions:

1. Promising future markets are characterized by a multi-objective decision space, where trade-off analyses must be conducted in early design phases, as they might highly impact the vehicles' size and configuration.

2. A rigorous and systematic methodology is needed that enables the exploration of a large combinatorial design space and supports quantitative trade-off analyses to facilitate the selection of a design.

3. Significant uncertainties originate from customer, regulatory, and market requirements. These uncertainties, which evolve throughout the design process and as the market grows, must be addressed to support the development of robust vehicles.

Based on these assertions, required capabilities to help decision makers address new markets have been identified and have led to the Research Objective: **to establish a methodology that enables a broad design space exploration at a conceptual level to select solutions against unclear objectives and under evolving uncertainty in requirements.** First attempts to address this objective using existing techniques encountered several difficulties due to the lack of methods equipped with the required capabilities. These gaps, identified in Chapter 2, gave rise to the two main Research Questions:

1. How can current conceptual design approaches be improved to enable a broader exploration of large and complex design spaces?

2. How can decision makers identify and prioritize a set of solutions robust to evolving uncertainty in requirements?

The literature review performed in this same chapter has enabled the formulation of hypotheses. To be validated, experiments have been developed and must be implemented following the approach described in the remaining of this chapter. This closes the first two steps of the generic top-down design decision support process as shown in Figure 28.

To develop the aforementioned methodology, there is a need for a test bed. Hence, suborbital tourism is selected and the proof-of-concept will be the design the design of a profitable, safe, and robust commercial suborbital program.

In order to address the research questions and hypotheses formulated in Chapter 2, a four-step approach is followed:

- Step 1: Establish the decision criteria

- Step 2: Define the design space
- Step 3: Evaluate alternatives
- Step 4: Make informed decisions

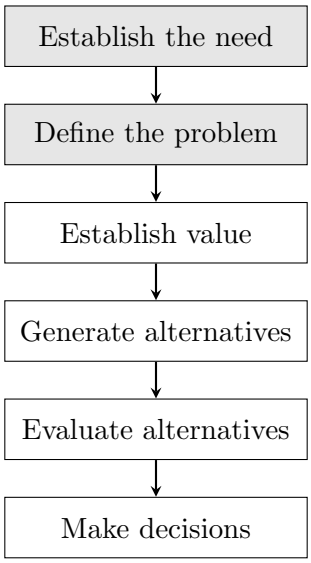



**Figure 28:** Generic top-down design decision support process

The objective of the following sections is to describe and detail each of these steps while also identifying the expected contributions.

## *3.1 Step 1: Establish the Decision Criteria*

The objectives of this first step are first to establish a set of decision criteria that will be used to compare and optimize the various alternatives, and second to address their implementation within the overall sizing and synthesis environment.

First, this step refers to Chapter 1 in order to identify the metrics of interest. These criteria can include economic or flying performance as well as customers' requirements such as safety and comfort. This step also aims at classifying these metrics into two distinct categories: constraints and optimization objectives. Constraints are parameters whose values have been fixed or bounded to limit the design freedom. Optimization objectives are parameters that must be optimized (maximized, minimized or be as close as possible to a

target value). Then, this step also addresses the implementation of such criteria. Indeed, it quantifies each criterion and provides a range of values that will be considered in the robustness analysis. The use of fuzzy logic for uncertainty modeling requires to model each noise variable by a membership function so that it can be further propagated through the optimization process. Once the objectives have been clearly defined and set, one can start to select the alternatives that will be considered. This task is addressed in the following section.

## *3.2 Step 2: Define the Design Space*

Defining the design space consists in determining all feasible alternatives that will be used for further comparison and optimization. The large diversity in concepts mentioned in Chapter 1 makes this task difficult. To address this challenge, a three-step process will be followed:

1. Generate all possible alternatives
2. Determine and describe the design variables of these alternatives
3. Generate all feasible architectures along with their specific design variables

This section aims at describing each phase of this three-step process.

### 3.2.1 Generation of Alternatives

In order to explore the entire design space, vehicles and missions are decomposed into features. Then, a literature review of all features will help list the available options for each feature. Once all options for all features have been listed, the conventional morphological matrix of the vehicle will be created.

### 3.2.2 Identification and Description of Design Variables

Based on the options previously generated, a deep literature review will be conducted to define the key drivers of each option. If no existing models exist at a conceptual design level, sensitivity analyses will be performed to extract the main design drivers. Hence, only the key design variables will be listed with their typical ranges.

### 3.2.3 Generation of All Feasible Architectures

The aforementioned morphological matrix does not take into account the compatibility between each alternative and optimization parameters. Hence, the methodology presented in Hypothesis 1.1 will be used to create the modified morphological matrix. First of all, options that are defined by the same design variables will be grouped in order to reduce the dimensions of the morphological matrix. Based on this new morphological matrix, a compatibility matrix will be created in order to support the generation of feasible architectures only. This final list of all feasible architectures, along with their corresponding design variables, corresponds to the definition of the design space. Figure 29 summarizes the process followed for design space definition.

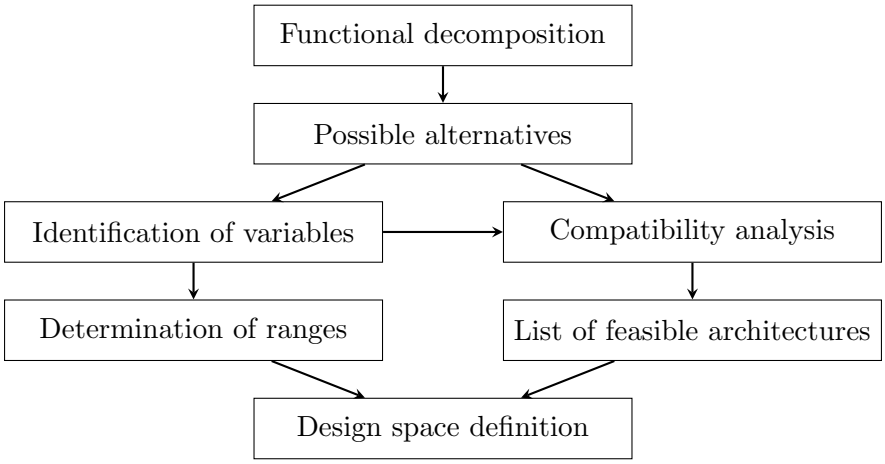



**Figure 29:** Methodology for design space definition

This systematic approach to generate feasible architectures attempts to provide a new methodology to explore the entire design space. The implementation of this step will enable the execution of Experiment 1.1 in order to validate Hypothesis 1.1.

Once each alternative has been characterized by its design variables, it can now be modeled and its performance can be evaluated. This performance evaluation requires a design framework, as discussed in the following section.

### *3.3 Step 3: Evaluate Alternatives*

This steps aims at evaluating alternatives against all objectives fast enough to enable further comparison and optimization. Many sizing and synthesis environments are already available. In order to find the most suitable for this application, selection criteria must be established. Hence, the performance evaluation environment must ideally benefit from the following capabilities:

- Automation: the environment must be able to be implemented within an automatic loop to execute several hundreds of thousands runs and therefore cover the entire design space and consider requirements' uncertainty. Thus, the environment must also be fast to run.

- Easy to learn: tools that require a long learning process will not be favored. Indeed, some software require an intensive and long training before being able to handle their operating principle and the syntax of their input and output files.

- Conceptual design level: the environment needs to include design variables commonly used at the conceptual level. Detailed information about the vehicle geometry are not available at this point of the process and therefore precise Computational Fluid Dynamics (CFD) calculations that require a detailed mesh of the vehicle cannot be included within the tool. If so, either the number of variables would become unmanageable or the execution time too large for a complete design space exploration at this phase of the design.

- Physics-based modeling: in addition to the architecture selection, the environment must be able to optimize a given architecture. If a delta wing is chosen, the tool must be able to optimize its shape in terms of sweep angle, taper ratio, root chord, etc. The main design parameters of the rocket engine must also be determined. Besides, the trajectory must be optimized in terms of flight path angle, speed, etc. This capability implicitly requires the use of physics-based modeling techniques in addition to, or instead of, historical data.

- Cost considerations: the integrated environment must be able to parametrically handle life-cycle costs so that each alternative can be evaluated not only in terms of performance, but also in terms of acquisition cost and operating costs. This is a critical enabler of the paradigm shift.

- Integration: the different modules (if more than one) must be integrated within a single environment. Moreover, this environment must have the capability of being integrated within an optimization environment.

Hence, this section starts with a comparison and evaluation of existing sizing and synthesis environments.

### 3.3.1 Review of Existing Sizing and Synthesis Tools

Marti and Nesrin Sarigul-Klijn present an overview of all possible methods for launch and recovery of manned suborbital vehicles [377]. To perform their study, they use historical data, first-order integration of the equations of motion as well as qualitative considerations in terms of safety, customer acceptance, and affordability. They finally provide a recommendation for the "best suborbital vehicle", which is, according to them, a winged body powered by a hybrid rocket engine that takes off vertically and lands horizontally like a glider (unpowered). While being quite exhaustive, their study does not follow a rational and systematic methodology since it is mainly based on the authors' expertise and several high-level and very generic equations. Moreover, all cost and safety considerations are qualitative and there is no optimization approach within each architecture. Nevertheless, they provide a quite exhaustive morphological matrix with an interesting overview of the corresponding existing concepts.

Rockwell Scientific developed Design Sheet, a framework that could help quantify the previous assertions. This Boeing proprietary tool allows conceptual designers to quickly explore design spaces with a large number of alternatives [46, 63, 355]. This is a general aerospace software in which designers can select their own Measure Of Effectiveness (MOE) among a variety of performance, survivability, responsiveness, and affordability criteria. It provides a series of plots that allow designers to easily perform multidisciplinary trade-off

studies. While being very flexible in terms of concepts and missions explored, this tool requires the users to input the required equations and is therefore based on very simple analytical models for each discipline. Therefore, an important amount of work is required from the users to be able to use the tool for their specific problems. The core of the tool lies in a series of optimization algorithms applied once the requirements, the constraints, and the models have been inputted. This tool has been used to compare Unmanned Aerial Vehicles (UAVs) [69], GA aircraft [63], and Reusable Launch Vehicles (RLVs) [46]. Finally, depending on the equations inputted, an architecture optimization could become relatively hard to perform due to the genericity of the tool.

A higher-fidelity tool was required to precisely design and study the performance of new aerospace systems. For this purpose, the AeroSpace Trajectory Optimization Software (ASTOS) was written by the German Aerospace Center (DLR) in collaboration with the Institute of Flight Mechanics and Control (IFR) at the University of Stuttgart. It was further improved by the European Space Agency and the development of the software is now conducted by Astos Solutions GmbH [477]. ASTOS is both a trajectory and a vehicle design optimization tool [23, 478]. It integrates aerodynamic and propulsion models, which enable an important variety of vehicles to be modeled: typical aircraft, launchers, and capsules. Detailed aerodynamic characteristics are provided as a function of the geometry, the angle of attack, and the flight point. A large variety of engines can also be modeled: throttleable and restartable rocket engines (liquid, solid, and hybrid) with variable specific impulse as well as various air-breathing engines. ASTOS also benefits from a library of existing aerospace vehicles that can be used as baselines. ASTOS is also able to handle two types of trajectory modeling approaches: collocation methods and multiple shooting methods. Users can specify multiple cost functions they want to optimize against as well as constraints on all types of variables such as maximum load factor or maximum temperature. ASTOS has already been successfully used for a series of applications: conventional launchers (Vega, Ariane 5 ECA, Soyuz), advanced launchers/RLV (Hopper, Skylon, Socrates), re-entry vehicles (Mars Demo Lander, X-38, Sphynx, ExoMars09), and orbit transfers/others (ConeXpress-OLEV, Lunar Excursion Vehicle) [474]. However, this software does not include any life-cycle cost

considerations. Finally, since it is very accurate, this software requires a very detailed input file with variables that are sometimes unknown or irrelevant at the conceptual stage of the design process. Thus, in addition to the longer execution time, it could lead to unexpectedly wrong results.

More specific design tools were developed to meet certain companies' needs. The Trajectory Synthesis Simulation Program (TSSP) was developed by the University of Oklahoma in collaboration with Rocketplane Limited, Inc. to support the development of its horizontal take-off and horizontal landing suborbital vehicle [214]. It is a multidisciplinary conceptual design level tool in which the main design disciplines (aerodynamics, propulsion, weight, thermal, performance, and trajectory) are coupled into an integrated environment. The optimizer uses the equations from the Energy State Approximation (ESA) technique detailed in Section 3.3.5. For a given vehicle, it tries to minimize the amount of fuel burned while satisfying the various constraints: maximum dynamic pressure, maximum load factor, maximum temperature, maximum heat transfer, etc. This tool provides a visualization of the multidisciplinary design space as well as a summary of the key mission performance. It also allows designers to check the sensitivity of the key design parameters on the overall vehicle performance. The tool is able to handle multiple geometries thanks to its aerodynamic module as well as multiple propulsion systems. However, it has only been developed for horizontal take-off and horizontal landing concepts. Moreover, it is not commercially available and does not include life-cycle costs within its framework.

J. D. Mattingly et al. presented an energy-based methodology for aircraft conceptual design [271]. The design process is based on a loop between constraint analysis and mission analysis. This loop relies on stated requirements and models for aerodynamics, propulsion, and weight. The objective of this optimization is to find the best design point (the one with the lowest thrust loading) that meets all constraints and mission requirements. It also ensures that the fuel available exactly matches the fuel required plus the safety reserves. The high-level representation of this methodology is displayed in Figure 30.

The final product of this process is a set of three parameters that globally describe the vehicle at a multidisciplinary level: the take-off gross weight $W_{TO}$ (mass/weight), the

surface area $S$ (aerodynamics), and the thrust at sea level $T_{SL}$ (propulsion).

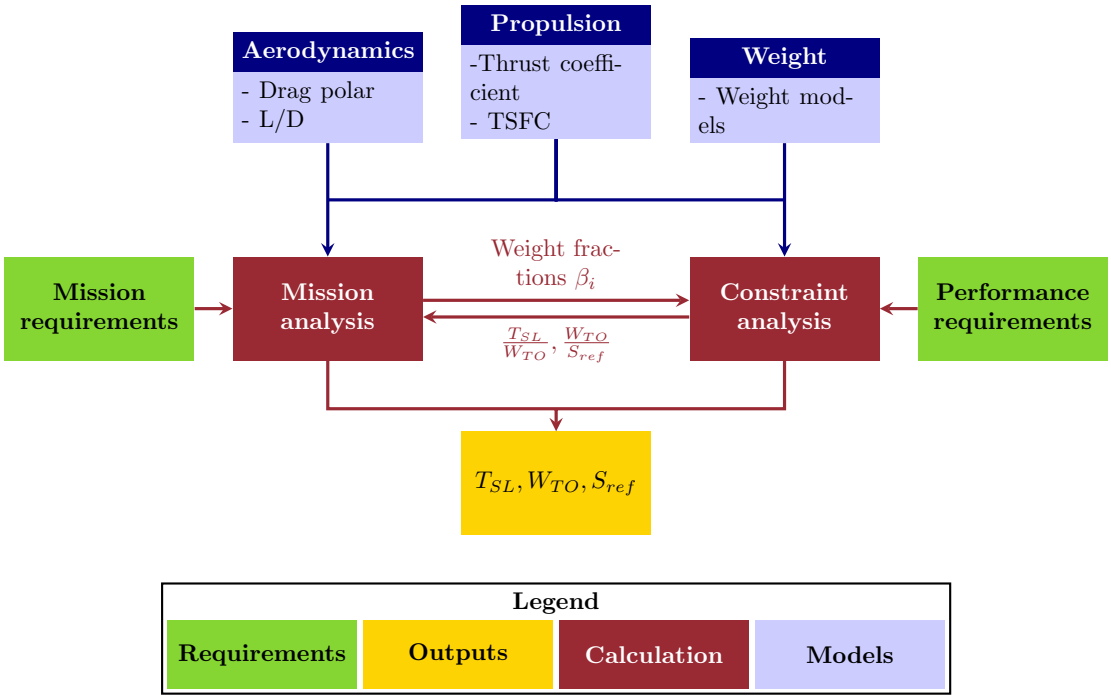



**Figure 30:** Overall design process [271]

The combination of NASA's FLight OPtimization System (FLOPS) and NASA's Aircraft Life-Cycle Cost Analysis (ALCCA) provides a conceptual design tool that enables both aircraft performance and life-cycle cost evaluation [307]. This combination is based on multidisciplinary modules: aerodynamics, propulsion, weight, performance, noise, emissions, and life-cycle costs. The aerodynamic module uses an improved version of Lockheed's Empirical Drag Estimation Technique (EDET). It decomposes the drag coefficient into four sources: skin friction, drag due to compressibility effects, induced drag, and pressure drag. While FLOPS allows the users to input their own drag polar, supersonic and hypersonic regimes are not well modeled. The propulsion module allows the users to choose between an internal performance model or their own engine decks. Nevertheless, FLOPS is not able to model rocket engines. Finally, the weight module is based on historical data and is corrected by different scaling factors that take into account variations due to material properties, flight regimes, etc. Hence, while this tool, which is publicly available, provides

a complete study at conceptual design level (including life-cycle costs), it cannot be used for hybrid configurations for which there is no historical data available. Indeed, it can only handle general aviation, civil transport, and fighter aircraft. Moreover, it requires a detailed geometric description of the vehicle which is unknown for this problem and could consequently lead to inaccurate results.

The Rapid Access-to-Space Analysis Code (RASAC) was developed by F. Villeneuve at the Aerospace Systems Design Laboratory (ASDL) at the Georgia Institute of Technology [454, 455]. This conceptual design tool enables the comparison of different launch vehicle architectures, their optimization as well as the evaluation of their robustness. The tool is divided into five disciplinary modules: trajectory, aerodynamics, propulsion, weight, and economics. The trajectory module calls four types of calculation methods, depending on the type of mission segment: Energy-State Approximation, two dimensional Newtonian calculation, Hohmann orbit transfer, and Breguet's range equation. The aerodynamic module is able to directly model all-body configurations or to use input tables or Response Surface Equations (RSEs). Therefore, except for all-body configurations, an additional tool is required to generate aerodynamic data (tables or RSEs) upstream from this aerodynamic module. The propulsion module enables the modeling of three types of engines: rocket, turbojet, and ramjet. It uses analytical equations but does not allow the user to distinguish between hybrid, liquid, and solid engines. The weight module relies on a set of equations that estimate the weight of 28 subsystems using geometric variables, take-off gross weight and empty weight estimations. Finally, the cost module relies on equations from the NASA-Air Force Cost Model (NAFCOM) and mainly uses the weight of each subsystem to predict the life-cycle costs. However, it does not support the comparison of the different rocket engine types.

Stanley et al. developed a methodology for the conceptual design of a two-stage rocket-powered reusable vehicle [414, 415]. The environment, capable of performing trade studies, is composed of five modules: geometry, aerodynamics, trajectory, aeroheating, and weight/-size. These modules take two sets of inputs: mission and propulsion. However, trade-studies require the execution of multiple configurations manually inputted. While life-cycle costs

are qualitatively discussed, no quantitative cost model is included in the environment.

J. R. Olds developed a framework based Multidisciplinary Design Optimization (MDO) for the conceptual design of new RBCC Single-Stage-To-Orbit (SSTO) launch vehicles [326]. The framework optimizes the vehicle for a single objective which is a minimum dry weight. Hence, it does not include any cost modeling modules. In addition, no conditions were implemented to enforce feasibility in the selection of design variables. As a consequence, around 30% of the designs generated by the analysis process were infeasible and the optimization under given requirements could take up to a day to run [244].

NASA Ames Research Center developed the Hypersonic Optimization Code (HAVOC), an integrated design environment for hypersonic launch vehicles [17, 235, 481]. The execution of the code results in a single vehicle, optimized against a given set of requirements. However, the modules are very specific to the study of hypersonic vehicles and cannot handle all suborbital configurations. Moreover, no cost considerations are included within the framework.

R. D. Braun developed Collaborative Optimization (CO), which is a new multi-level optimization technique that distributes the overall optimization to several disciplinary subproblems [49]. CO has then been applied to SSTO launch vehicles [50, 51]. Using 95 design variables and 16 constraints, the algorithm uses optimization techniques based on gradient calculation and cannot handle discrete variables. Moreover, the tools used require an execution time too long for robustness analysis [244].

The environments discussed in this section are compared and evaluated in Table 11. For each criterion, the rating is as follows: a double check mark if it perfectly fulfills the criterion, a single check mark if it partially fulfills it, and nothing if it does not fulfill at all. This rating will be used for the remaining of this chapter.

**Table 11:** Comparison of the various sizing and synthesis codes

| | Fast | Available | Easy | Design space exploration | Conceptual level | Architecture optimization | Cost modeling | Automation |
|---|---|---|---|---|---|---|---|---|
| **Sarigul** | | ✓✓ | ✓✓ | ✓✓ | | | | |
| **Design Sheet** | ✓✓ | | | ✓✓ | ✓ | ✓ | ✓ | ✓✓ |
| **TSSP** | ✓✓ | | ✓ | | ✓ | ✓✓ | | ✓ |
| **Mattingly** | ✓✓ | ✓✓ | ✓✓ | | ✓ | | | ✓✓ |
| **FLOPS** | ✓✓ | ✓✓ | ✓ | | ✓ | ✓✓ | ✓✓ | ✓✓ |
| **ASTOS** | ✓ | ✓ | | ✓✓ | | ✓ | | ✓ |
| **RASAC** | ✓✓ | ✓✓ | ✓ | | ✓✓ | ✓ | ✓ | ✓✓ |
| **Stanley** | ✓✓ | | ✓ | | ✓✓ | ✓ | | |
| **Olds** | | ✓ | ✓ | | ✓ | ✓✓ | | ✓ |
| **HAVOC** | ✓✓ | | ✓ | | ✓ | ✓✓ | | ✓✓ |
| **Braun** | | | | ✓ | ✓ | ✓✓ | ✓✓ | ✓ |

This evaluation shows that there is a lack of readily available sizing and synthesis environment that can evaluate all various alternatives against life-cycle costs. This leads to the following Research Question:

**RESEARCH QUESTION 3: What are the key enablers of a sizing and synthesis environment able to evaluate the various alternatives of suborbital vehicles for further comparison and optimization at a conceptual design level?**

The previous analysis and comparison showed a common thread to all quantitative sizing and synthesis environments: a decomposition into several disciplinary modules. Each of these modules must be capable of modeling the various alternatives. Moreover, a decomposition into disciplinary modules is also justified by the various alternatives identified for each feature in Section 1.2. In particular, when designing unconventional configurations, disciplines that describe the problem, as well as their interactions, must be integrated [279].

While all environments benefit from performance modules (weight, aerodynamics, propulsion, and trajectory), economics and safety are not always included. However, as discussed in Section 1.1, this is a significant required capability that must be included in the design framework. In order to articulate all these modules, different structures might be used. They are categorized based on the number of optimization levels: single-level, bi-level, and multi-level [171].

Single-level structures are based around one centralized optimizer and distribute the analysis to different analyzers. Three main formulations were identified: All-At-Once (AAO), Individual Discipline Feasible (IDF), and Multi-Disciplinary Feasible (MDF) [98]. In AAO approaches, all disciplinary variables are optimization variables and disciplinary equations correspond to optimization constraints. Thus, designs are only consistent at convergence. IDF approaches use the same principle but ensure disciplinary feasibility at each iteration. Finally, MDF approaches only output consistent multidisciplinary designs at each iteration. This consistency has a cost in terms of number of function calls so that the time required to generate a design increases from AAO to IDF and to MDF. The number of design variables that must be controlled by the optimizer decreases from AAO to IDF to

MDF and thus the optimization problem becomes smaller for MDF approaches. On the one hand, MDF approaches require the development of efficient system analysis tools, usually based on the Fixed-Point Iteration (FPI) method. On the other hand, AAO cannot directly use disciplinary tools with their corresponding equations since they have to be set as design constraints. Finally, it has been shown by Allison [12] that MDF algorithms perform well for low coupled problems while IDF is relatively insensitive to coupling. While MDF ensures consistency, IDF ensures feasibility at each iteration by enforcing the various constraints.

To match the increasing complexity of engineering problems, multi-level optimization strategies have also been developed. They better match typical organizational structures and enable the overall optimizer to distribute the tasks to multiple sub-optimizers. Among them, two representatives of these methods are CO and Analytic Target Cascading (ATC). CO is a bi-level approach that matches problems described by disciplines at the same hierarchical level while ATC is a multi-level approach with hierarchical rules and feedforward communication. CO promotes subspace autonomy by allowing each discipline to benefit from its own optimizer. Compared to ATC, this approach has weak interactions between disciplines, which only receive information from the main optimizer. CO only requires one execution of the main problem and is based on shared optimization while ATC requires multiple executions and is based on shared coordination. Finally, CO seems to be more efficient in terms of function calls than ATC. While being useful for large problems within large organizations, these multi-level approaches do not correspond to conceptual level characterized by a relatively limited number of design variables. Table 12 summarizes the capabilities of each optimization structure. The overall problem contains both continuous and discrete variables so that a meta-heuristic algorithm will probably be used for the global optimization. Hence, the number of function calls is large and increases with the number of variables. In addition, the number of function calls can be further decreased by avoiding inconsistent solutions to be generated. Since a conceptual design approach is followed, the number of design variables is not too large (around 30 variables are expected). Therefore, an MDF approach is preferred [112].

**Table 12:** Comparison of the various optimization frameworks

| | MDF | IDF | CO | ATC |
|---|---|---|---|---|
| **Consistent design only** | ✓✓ | | | ✓✓ |
| **Good convergence** | ✓ | ✓✓ | ✓ | ✓ |
| **Easy to implement** | ✓✓ | ✓ | | |
| **Good for many variables** | | | ✓✓ | ✓✓ |
| **Feasible design only** | | ✓✓ | ✓✓ | |

However, with a traditional MDF process, solutions are not necessary feasible before the final convergence since constraints are handled by the optimizer. In addition, there is a need for decreasing the number of design variables in order to increase the efficiency of the algorithm. As discussed by Defoort et al. [112], local coupling is a commonly used method to reduce both the number of variables and the strength of the coupling in the overall design framework. Defoort suggests the use of FPI, but depending on the problem, FPI convergence may not occur or may produce a suboptimal solution [171]. Hence, it will be replaced by a local optimizer using gradient-based methods. The idea is to generate geometries and ensure that engines are optimally sized to meet trajectory requirements. As a consequence, based on the geometry generated by the optimizer, its weight and aerodynamic characteristics will be sequentially computed. There is no feedback between aerodynamics and weight. Then, an optimizer including both the trajectory and the propulsion modules will be created. It will include the highly coupled trajectory and propulsion modules to find the optimized propulsion system that enables the vehicle to perform the minimum fuel-to-climb trajectory. The only feedback will be the change in weight/size due to the new propulsion system. The implementation of the local optimizer will thus reduce the strength of the coupling and enable the enforcement of various constraints so that only feasible solutions will be generated. Finally, since the intermediate optimizer is built around continuous variables, it will be fast to run and therefore accelerate the overall optimization process by using more efficient gradient-based algorithms for a given group of variables. The proposed structure is presented in Figure 31 and is characterized by a modified MDF approach. Hence, in addition to only generate consistent designs, this MDF approach will be

improved by a local optimizer which will also ensure feasibility among the generated designs.

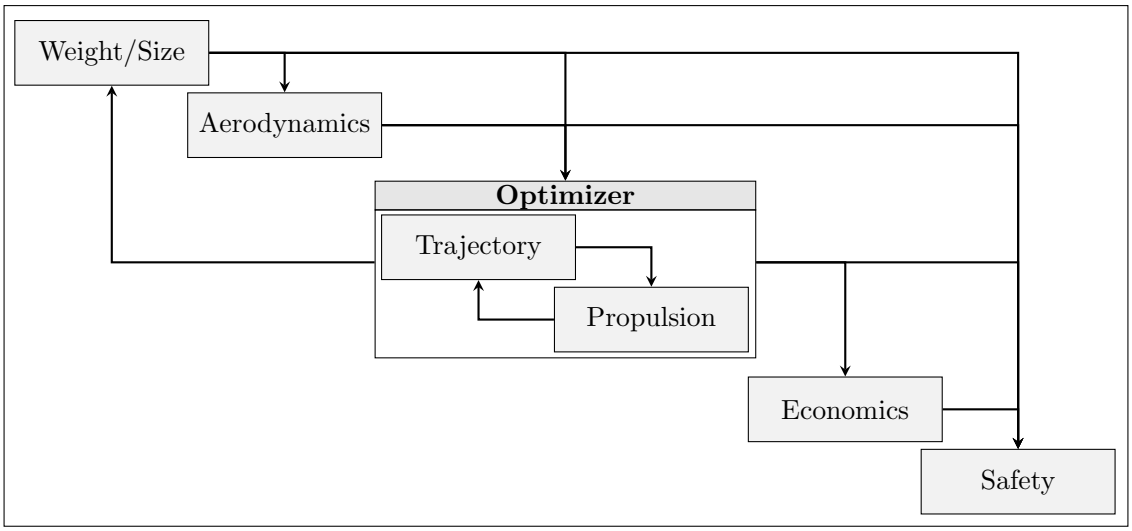



**Figure 31:** Proposed structure of the design environment

To enable a successful use of this structure, the execution of the different modules needs to be fast. Figure 32 notionally represents the characteristics of different modeling techniques in terms of accuracy and number of variables considered.

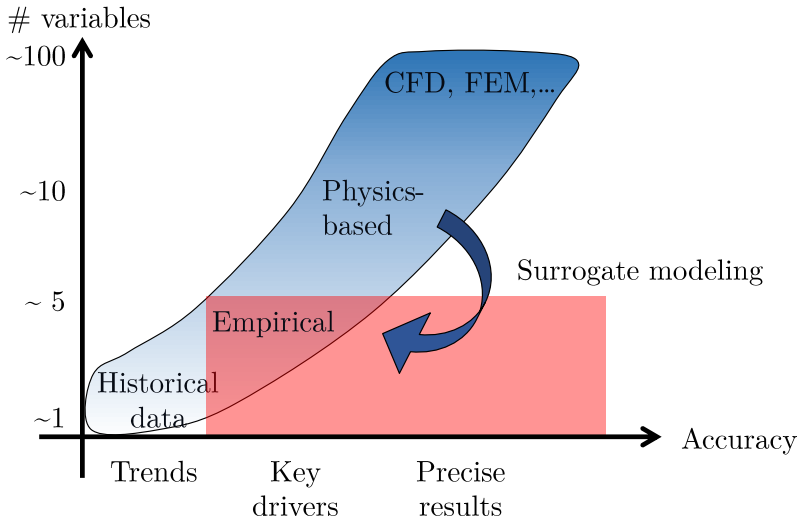



**Figure 32:** Characteristics of the different modeling techniques

The area that must be targeted in this study is displayed in red on the graph. As shown, when available, empirical relations can be used. However, since some concepts are new, there is a lack of historical data to build these relations, and consequently surrogate modeling can be used to speed up the process. Surrogate modeling takes advantage of simple analytical equations to approximate complex physic processes. The different surrogate modeling techniques include Response Surface Methodology (RSM) [306], Artificial Neural Network (ANN) [82], and Kriging [269]. These models, while ensuring a good accuracy would allow a real-time execution of the environment. Hence, both empirical relations and surrogate modeling of physics-based models can be leveraged to build the disciplinary modules.

These observations lead to the formulation of the following Hypothesis.

**HYPOTHESIS 3: IF a sizing and synthesis environment based on a modified Multi-Disciplinary Feasible (MDF) approach and using both empirical relations and surrogate models is created THEN performance of the various alternatives of suborbital vehicles can be evaluated for further comparison and optimization.**

In order to validate this Hypothesis, Experiment 3 is implemented. As inputs, it requires a description of existing vehicles and their corresponding missions. In addition, it also requires each discipline to be modeled by empirical relations or surrogate models. Then, the design framework will be created, as described in Figure 31 and executed for the aforementioned vehicles. To fully validate Hypothesis 3, three elements need to be validated: accuracy (to match actual data and ensure consistent results), usability (to provide the multi-disciplinary metrics for all types of suborbital vehicles), and speed (to enable further comparison and optimization). This will be done through four questions:

1. Are the calculated vehicles' performance consistent with the literature?

2. Is the execution time acceptable (around 10 seconds)?

3. Does the modeling and simulation environment provide all the metrics required to get a complete picture of the vehicle?

4. Is the modeling and simulation environment capable of evaluating all types of suborbital vehicles?

Experiment 3 requires models for each discipline that are fast to run and accurate. The different disciplinary modules with their following capabilities are listed below:

- Weight modeling for new concepts including all airframe configurations and propulsion types.
- Propulsion modeling for both air-breathing engines (turbofans, turbojets, etc.) and rocket engines (liquid, solid and hybrid propellants).
- Aerodynamic modeling for multiple configurations such as typical fixed-wing bodies, launchers, etc.
- Trajectory optimization based on data from the first three disciplines and that can handle various types of missions.
- Life-cycle costs assessment (acquisition cost, operating cost, etc.).
- Safety evaluation using quantitative metrics common to all suborbital vehicles.

A review and evaluation of the available approaches and tools for each discipline is provided in the following sections. The goal of this review is to help identify potential candidates for inclusion in the multidisciplinary environment as well as gaps in existing capabilities.

### 3.3.2 Weight and Size Estimation

Weight reduction has traditionally been the ultimate goal of designers since it is directly associated with fuel efficiency and consequently cost efficiency. Nevertheless, even if many studies have shown that affordability is not necessarily obtained for minimum weight, weight estimation still remains a significant task in aircraft and spacecraft design. While weight

estimation seems relatively easy to do at the end of the design process by simply summing the weight of all components, it turns out to be extremely complex in early design stages. Due to specific material properties, there is also a strong relationship between weight and dimensions. Therefore, an accurate weight estimation in early phases also implies an accurate vehicle description. Three main approaches can be distinguished: the Fixed-Fraction method, the Statistical Correlation method, and the Point Stress Analysis method [97]. Each of them will be described and investigated in order to identify suitable existing tools and models. They will be compared and evaluated in order to identify the ones that can be used to build the weight module.

#### *3.3.2.1 Weight Estimation Methods*

The three aforementioned techniques for vehicle weight estimation have their specific level of complexity, accuracy, and detail that must be understood for a wise selection.

The Fixed-Fraction method is the simplest, fastest, and easiest method since it assumes that the weight of a given component is a fraction of the empty weight, the take-off gross weight or the wing area [354, 368]. Equation 26 provides a typical example of the wing weight estimation $W_w$ as a function of its exposed planform area $S_e$ in imperial units [354].

$$W_w = 10 \times S_e \tag{26}$$

However, this method has some drawbacks because it assumes that there is only one key parameter to estimate the weight of a component. For example, using this approach, the wing described in Equation 26 would have the same weight independently of its sweep angle, taper ratio, and material. Moreover, it does not account for the location of the landing gear and engines which have a significant impact on the weight of the wing. As such, it can only be applied to one specific type of vehicles. Even if it could be very efficient to predict the weight of a derivative aircraft, it cannot be used for innovative or unconventional vehicles. Besides, this estimation method cannot be used to geometrically optimize a configuration due to its small number of input parameters.

The Statistical Correlation method uses historical data to develop empirical equations.

It is a generalization of the first method using more complex relationships with more parameters. Experience shows that a power model (weight as a function of the power of some variables) often provides good correlations [97]. Moreover, the parameters of this model can be easily estimated because there is a linear relationship between the logarithm of the weight and the variable. The general formula is given in Equation 27, where $\alpha_i$ and $\beta_i$ are two coefficients found empirically, $X_i$ the parameters, and $W$ the component weight. This type of relation is often called Mass Estimating Relationship (MER). Some implementations of this technique are iterative since a component weight could use another weight to be determined.

$$W = \sum_i \alpha_i X_i^{\beta_i} \tag{27}$$

This technique overcomes most of the pitfalls previously described due to its ability to take into account the impact of more than one key parameter on the component weight. This method is also easy to implement and very fast since it uses on analytical relations. By only taking into account key factors, these equations are usually perfectly adapted to conceptual level considerations. However, one of the main drawbacks of this method is its strong reliance on historical data so that difficulties could arise when considering enhancements in materials as well as new or hybrid configurations. This could partially be overcome by introducing correction/calibration factors for the different materials, levels of technology, etc. Examples of successful implementations of this approach are numerous and it has become the most widely used method in the aerospace domain [174, 354, 375].

The Point Stress Analysis method is a physics-based method that uses detailed information about the material and shape of the components. This method can only be applied to the main mechanical components such as the wing and the fuselage since there is no existing mathematical prediction techniques for other subsystems such as avionics, hydraulics, etc. Indeed, the method sizes these components based on specified loads they must be able to carry. Due to the complexity of the problem, a resource and time-consuming computer program is needed. Examples of such methods are the Structural WEight Estimation Program (SWEEP) [20], Patran/Nastran [302, 303], Abaqus [105, 399], etc. This method also requires very detailed information about the vehicles and the loads. As such, this technique

is not suitable for conceptual design analyses.

Based on this analysis, the Statistical Correlation method will be used for this work due to its simplicity, availability, and optimal level of detail. The most common implementations of this approach are presented, compared, and evaluated below.

#### *3.3.2.2 Existing Weight Estimation Models*

The Weight/Volume (WTVOL) module of the Space Shuttle Synthesis Program (SSSP) was developed by NASA to predict the weight and the dimensions of each component of the space shuttles [96]. It consists of a series of equations that use existing weight data and specific inputs for the thermal protection and propulsion systems. However, this program was built for the Space Shuttle and consequently has very limited applications.

The Weight Analysis of Advanced Transportation Systems (WAATS) computer program has been developed by the Aerophysics Research Corporation for NASA [174]. It uses power law formulas to predict the weight of a large number of subsystems. The tool assumes that the propellant weight and the physical characteristics are known. An iterative approach is used until convergence on the take-off gross weight is reached. Propulsion systems such as turbojet, ramjet, and liquid rocket engines are investigated. Different regression parameters are proposed depending on the type of vehicle: low speed vs. high speed.

Hypersonic Aerospace Sizing Analysis (HASA) is a model that predicts the vehicle length and volume consistent with body, fuel, structural, and payload weights [194]. It was developed by Sverdrup Technology and can handle SSTO and Two-Stage-To-Orbit (TSTO) vehicles as well as hypersonic and supersonic aircraft. This model takes into account propulsion and material enhancements. In addition to air-breathing engines, liquid rocket engines are modeled with various propellants. Results show an accuracy of $\pm 10\%$ in terms of weight and size.

D. P. Raymer provides a summary of MERs used by current airframe companies [354]. Applications of his equations include all types of aircraft: GA, fighter, cargo, etc. The equations come from various sources and have been adapted by including "fudge factors" [220, 419, 420]. The equations capture almost all aircraft components and are perfectly

adapted to optimization at a conceptual design level.

M. H. Sadraey created a set of equations for estimating the weight of aircraft's main components [375]. These equations can be applied to GA, transport, cargo, and fighter aircraft. He uses a hybrid method that combines material density, published data about current aircraft, his own empirical relations and relations from existing studies [381, 419, 420, 441]. However, M. H. Sadraey only provides equations for lifting surfaces, fuselages, engines, and fuel systems.

The RASAC has a weight estimation module that decomposes the vehicle into 28 elements [455]. The weight is then given as a function of the vehicle geometry, propellant properties, vehicle take-off gross weight, and other key design drivers. RASAC uses MERs from various sources in order to create a complete weight module for launch vehicles [174, 194, 366].

#### *3.3.2.3 Model Selection*

Table 13 compares the various weight estimation models. The two best candidates are RASAC and HASA. Hence, inspired by RASAC, a weight module will be developed using MERs from different sources in order to benefit from a customized module with the required flexibility and an appropriate level of detail. However, none of these models is capable of appropriately modeling the three rocket engines to support their optimization.

**Table 13:** Comparison of the various weight estimation models

| | **Winged-body** | **Non winged-body** | **3 rocket engines** | **Propulsion optimization** | **Appropriate level of details** |
|---|---|---|---|---|---|
| **WTVOL** | ✓ | | | | ✓✓ |
| **WAATS** | ✓✓ | ✓ | | ✓ | ✓ |
| **HASA** | ✓✓ | ✓✓ | | ✓ | ✓✓ |
| **Raymer** | ✓✓ | | | | ✓✓ |
| **Sadraey** | ✓✓ | | | | ✓✓ |
| **RASAC** | ✓✓ | ✓✓ | | ✓ | ✓ |

The propulsion system (engines and propellants) also represents the biggest part of the

vehicle weight [157, 377]. Besides, the overall vehicle's weight and performance are highly sensitive to propulsion choices so that a thorough study of the propulsion system will be performed in the following section. In addition, hybrid rocket propulsion is a new propulsion type and the lack of historical data requires the use of a physics-based modeling for weight and performance estimation. This leads to the following Research Question:

**RESEARCH QUESTION 3.1: What modeling techniques can be leveraged to evaluate both weight and performance of liquid, solid, and hybrid rocket engines at a conceptual design level for further comparison and optimization?**

According to the previous Research Question, special attention should be paid to propulsion modeling. Hence, propulsion modeling will be discussed in the following section.

### 3.3.3 Propulsion Modeling

Since a suborbital vehicle travels across all atmospheric layers, it must evolve within low and high-density air. Figure 33 represents a high-level decomposition of the different available propulsion types. Each of them has its own characteristics, applications, advantages, and drawbacks due to its specific operating principle. The purpose of this section is to study each of them in order to find the possible candidates for suborbital vehicles. Once they have been selected, their design methodology will be presented based on existing approaches.

An overview of the main features helps highlight the potential applications for each technology within the suborbital flight environment. Indeed, suborbital vehicles might be powered by up to three propulsion systems with different requirements and purposes:

- Main engine: it must deliver a huge amount of thrust during a short period of time in order to provide sufficient speed for the vehicle to reach at least 100 km. Since this engine must be able to work at high altitude, where the air is extremely thin, a rocket engine has to be used.

- Auxiliary engine: depending on the launch type, an auxiliary engine may be installed for low speed and low altitude segments. This engine, which only helps the vehicle to

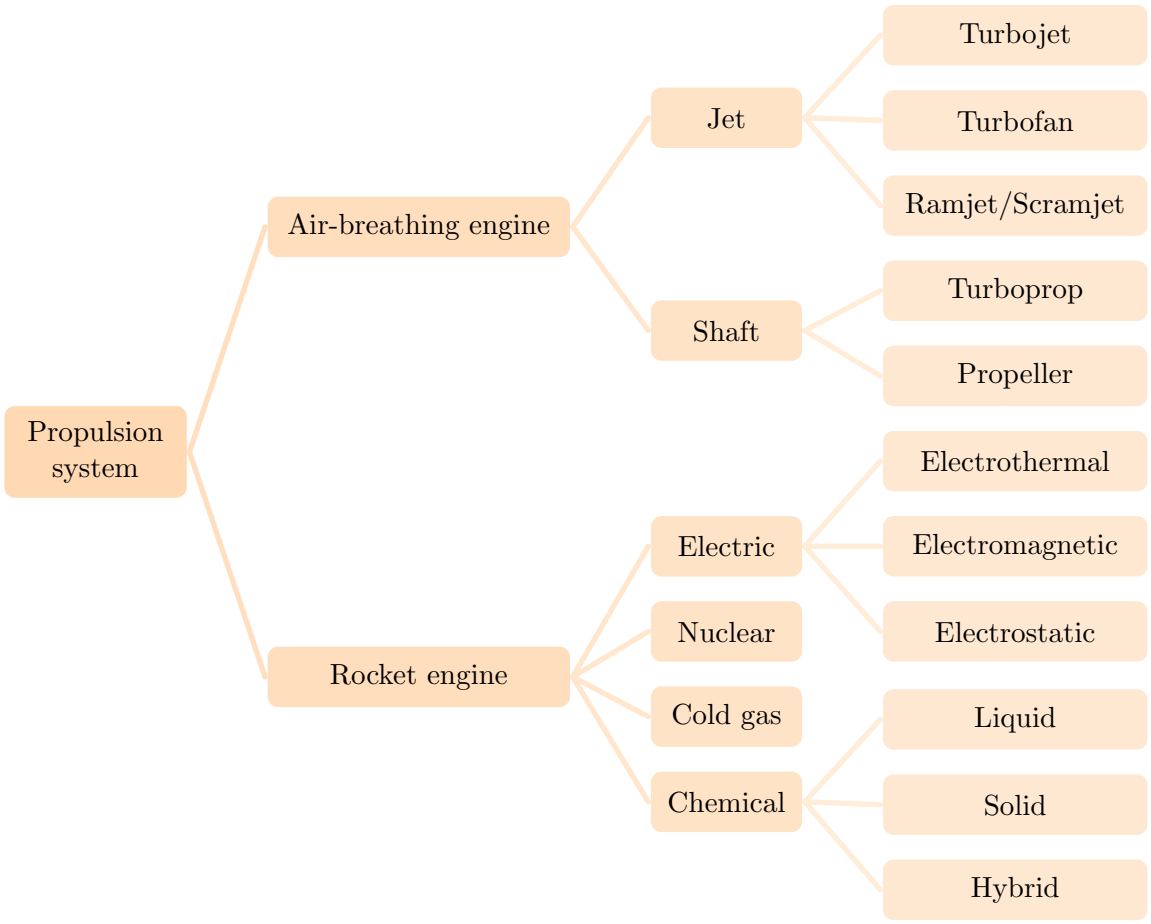



**Figure 33:** Propulsion alternatives

reach an intermediate altitude, where the main engine is started, requires less thrust than the main engine. Consequently, air-breathing engines will be preferred due to their higher efficiency.

- Attitude control system: at 100 km, the atmosphere is too thin to allow the use of aerodynamic surfaces (aileron, rudder, and elevator) to control the orientation of the vehicle. Therefore, low thrust and short duration rocket engines must also be installed. This reaction control system should have a thrust of approximately 100 N and must be very fast to react.

#### *3.3.3.1 Overview of the Different Alternatives*

**Shaft engines:** Shaft engines can be split into two different categories: reciprocating engines (or propeller) and turbine-powered engines such as turboprops. Propeller engines are piston engines that drive a rotating crankshaft in order to rotate a propeller. Turboprop also rotates a propeller but it is driven by a gas turbine instead of a piston engine. These engines are both very efficient but provide less thrust than turbojet or turbofan engines. Moreover, their optimum design Mach number is below 0.4 for a reciprocating engine and around 0.6 for a turboprop. In addition, since they require high-density air to produce thrust, they are not efficient at high altitude.

**Jet engines:** A turbojet engine is composed of five main components: diffuser, compressor, combustion chamber (burner), turbine, and nozzle (Figure 34). The compressor and the turbine are mechanically linked by a crankshaft. The incoming air is decelerated by the diffuser, compressed and heated by the compressor in order to reach optimal combustion conditions. Finally, the flow is expanded by the turbine and ejected throughout the nozzle.

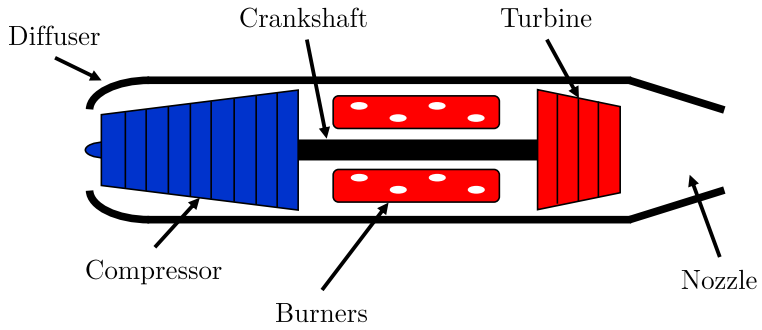



**Figure 34:** Notional turbojet engine [437]

The amount of thrust generated $T$ mainly depends on the difference between the exhaust speed $V_j$ and the free-stream velocity $V_\infty$ as well as the mass flow rate $\dot{m}$. Equation 28 presents the simplified thrust equation. The principle of a turbofan is the same except that there is a bypass flow which does not go through the engine core.

$$T = \dot{m}\,(V_j - V_\infty) \tag{28}$$

Ramjet and scramjet engines are simpler since they do not have moving parts. The compression is achieved only by using the speed of the incoming air. Consequently, they are only composed of a diffuser, a burner, and a nozzle. Figure 35 compares the main types of jet engines in terms of efficiency (Isp) and design Mach number. One can conclude that each type has its own operating Mach number range, which must be considered during the engine selection.

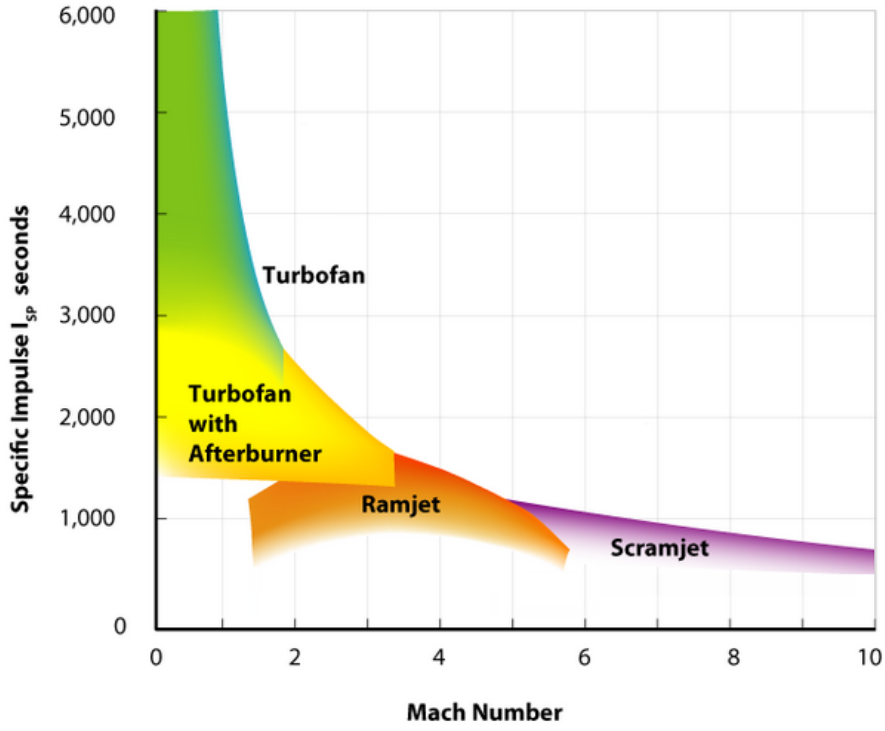



**Figure 35:** Comparison of the different jet propulsion types [186]

**Chemical rocket engines:** For rocket engines, the thrust is produced by accelerating the fluid exhaust without using the incoming air. An exothermic chemical reaction of the propellants within the combustion chamber heats the fluid, which is then expanded (and accelerated) through a supersonic propelling nozzle. Chemical rocket engines can be classified into three main categories depending on the nature of the propellants: liquid, solid, and hybrid. These engines are the most widely used among existing launchers and suborbital concepts. The three aforementioned types can mainly be distinguished based on

the following specific characteristics:

- Liquid propulsion: propellants are stored in liquid form into two separated tanks. The specific impulse depends on the nature of the propellants but is usually between 250 and 500 s.

- Solid propulsion: the oxidizer and the reducer are both mixed into a single combustion chamber in solid form. Even if it can reach high thrust, the specific impulse usually remains below 270 s.

- Hybrid propulsion: this type of propulsion combines the liquid and the solid characteristics since the reducer is usually stored in solid form while the oxidizer is separately stored in liquid form. The specific impulse is usually between 290 and 350 s.

Even if liquid propulsion was predominantly used on the existing vehicles, all of them must be investigated due to their competing advantages and drawbacks. The latter are qualitatively exposed for each propulsion system and must be considered in the design process [189, 216, 229, 423, 445].

Liquid propulsion engines have high specific impulse and can be easily controlled, stopped, and restarted. Most liquid propellants are environmentally friendly and maintenance is easy on this type of engines. Nevertheless, the design is usually complex, pumps must be added and the tanks must be pressurized by an auxiliary system. Finally, the density of the fuel is relatively low and the risk of leaks or spills is high.

Solid propulsion engines have simpler design and are easy to operate without any leaks or spills risks. Besides, solid propellants have a relative high density. However, they are almost impossible to control, stop, restart or reuse once they have been ignited. Consequently, they cannot be tested. In addition to having small specific impulse, their propellants are usually very polluting.

Compared to solid propulsion, hybrid propulsion has higher specific impulse, it is safer, less toxic, more controllable, and easier to operate. Compared to liquid propulsion, it is mechanically simpler and safer. While being denser than liquid propellants, hybrid propellants are convenient for including additives. In addition, hybrid propulsion engines are expected

to be cheaper than the other two types in terms of both RDT&E costs and recurring costs. However, they also have some drawbacks: their regression rate is usually too slow to produce very high thrust and for a given oxidizer rate, the oxidizer-to-fuel ratio shifts. These off-peak operations, combined with possible unburned residues, make them less efficient.

**Electric engines:** Electric propulsion uses electrical power to accelerate a propellant and consequently extract thrust. Three main categories of electric engines emerge based on the way the propellant is accelerated:

- Electrothermal: the propellant is heated by an external source of energy.
- Electromagnetic: the Lorentz force is used to "push" the gas and therefore produce thrust.
- Electrostatic: a high voltage electrostatic field is used to directly accelerate ions.

In general, electric propulsion provides high performance by increasing the specific energy of the propellant stream. Thus, they can achieve higher exhaust velocities which then enable lower propellant mass and also greater payload-to-mass ratio. However, electric propulsion is said to be power limited since the rate at which the external source can supply energy to the propellant is constrained by the mass available for the power system. Therefore, even if these technologies appear to be very efficient and promising, the available thrust is very small (usually lower than 10 N).

**Nuclear engines:** A nuclear rocket engine heats a working fluid using nuclear fission instead of typical chemical processes. Nuclear engines can deliver a huge amount of thrust with a high specific impulse compared to chemical ones [59]. J. Ramsthaler compares two 65-kN engines and concludes that nuclear engines have a specific impulse twice as big as the chemical ones as well as a longer life time. Nevertheless, nuclear engines have some drawbacks which disqualify their potential use. Indeed, in addition to their high structure weight and high cost, they present a large number of safety issues [88]. The protection of crew members from radiations appears to be heavy and expensive. Moreover,

all maintenance procedures must be changed and personnel must be protected. Finally, a problem during the flight could lead to a large and disastrous contamination [58].

**Cold gas engines:** The energy is pre-stored through a high pressure gas. Usually, the gas compressed in the tanks is $N_2$, $H_2$ or He. Then, the gas is released through a feed system and accelerates in a converging-diverging nozzle. Therefore, this system is simple, safe, and easy to operate but can only provide low thrust (usually less than 100 N) compared to chemical rocket engines.

#### *3.3.3.2 Engine Selection*

A summary of the main features of each technology is provided in Tables 14 and 15. An order of magnitude of the specific impulse (Isp) or Thrust Specific Fuel Consumption (TSFC) and the available thrust or specific thrust is displayed as well as the main advantages and disadvantages. For comparison purposes, air-breathing engines and rocket engines have been separated.

**Table 14:** Characteristics of the different rocket engines [216]

| | **Isp (s)** | **Thrust (N)** | **Advantages** | **Disadvantages** |
|---|---|---|---|---|
| **Cold Gas** | 60-250 | 0.1-50 | simple, safe, low contamination | low Isp |
| **Chemical** | 150-350 | 0.1-$10^7$ | high thrust, widely used | moderate performance, safety concerns, combustion issues |
| **Nuclear** | 800-1,000 | up to $10^7$ | high Isp, high thrust | unproven, low thrust loading, expensive, safety concerns |
| **Electric** | 500-10,000 | $10^{-3}$-20 | huge Isp | heavy system, low thrust, limited experience |

Electric engines will not be considered due to their lack of applications for suborbital vehicles (no low-thrust engines are required). Since nuclear engines are politically controversial and hazardous, they will not be considered either. Cold gas engines can only be used for attitude control due to its small thrust. All three chemical rocket engines could serve

as main rocket engines while only liquid chemical rocket engines can be used for attitude control purposes. Indeed, solid engines are difficult to restart and hybrid engines are difficult to precisely control. Finally, since shaft engines can only be used at very low altitude and only benefit from a small specific thrust, they will not be modeled.

**Table 15:** Characteristics of the different air-breathing engines [283, 354]

| | **TSFC ($hr^{-1}$)** | **Specific thrust (lb/lb/sec)** | **Advantages** | **Disadvantages** |
|---|---|---|---|---|
| **Ramjet** | 1-4 | 100-250 | light, high specific thrust, low cost | inefficient off-design |
| **Turbojet** | 0.8-0.9 | 70-90 | light, high specific thrust, high altitude | high TSFC |
| **Turbofan** | 0.4-0.7 | 20-70 | small TSFC | bulky, heavy |
| **Shaft** | 0.2-0.3 | 3-10 | very small TSFC | low speed and very low altitude only |

Ramjets and scramjets are only very efficient at their design supersonic Mach number. However, since the auxiliary power source is only used to accelerate the vehicle, it principally operates at subsonic speeds so that they will not be considered in this work. Nevertheless, both turbojets and turbofans with or without afterburners are potential alternatives for the auxiliary source of thrust. The resulting morphological matrix for the propulsion system is illustrated in Table 16.

**Table 16:** Morphological matrix for the propulsion systems

| **Alternatives** / **Functions** | **Alt. 1** | **Alt. 2** | **Alt. 3** |
|---|---|---|---|
| **Main engine** | Liquid | Solid | Hybrid |
| **Auxiliary engine** | Turbojet | Turbofan | None |
| **Attitude control** | Cold gas | Liquid | |

#### *3.3.3.3 Propulsion Design Process*

The general design process of an engine can be divided into four main steps and is described in Figure 36. First, all the requirements must be gathered (operational environment, performance, mission, constraints, etc.). Based on the requirements, design choices including cycle type, cooling approach, and materials must be made. Based on those characteristics, performance must be computed following by a weight estimation. Finally, a loop must be created in order to verify that the requirements are met and optimize the engine by modifying some design parameters. The requirements are defined by the trajectory in terms of thrust required, duration of the combustion, etc. The main design choices are handled by an optimizer.

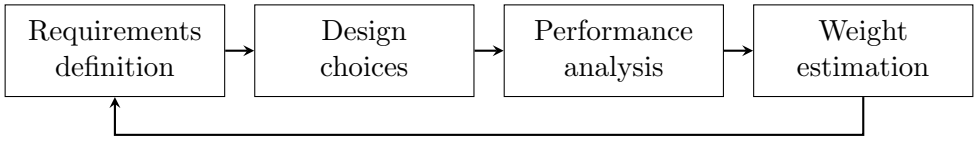



**Figure 36:** General engine design process

For each propulsion type (air-breathing and rocket engines), a tool must be developed that has both performance analysis and weight estimation capabilities. The following provides a brief review of the existing tools with respect to each of these capabilities.

**Air-breathing engines:** Air-breathing engines are well-known and widely used today. Moreover, since these engines equip a majority of current aircraft, many tools and databases are available to assist engineers in their design. However, in addition to being proprietary, these very accurate software are based on high-fidelity analyses and CFD codes. They require a substantial amount of data to be run and were often developed in-house based on companies' own design practices.

There are three main approaches to evaluate both performance and weight of a jet engine: a whole engine approach, a component-based approach, and a cycle parameters-based approach. They must be examined and compared in order to find the one which best meets the following criteria: fast to run for design space exploration, capture variables

used at the conceptual design level, and accurate enough to enable the comparison between different engine architectures and cycles. The aforementioned approaches are described and compared below:

- Single parameter approach: in this approach, the weight or the main performance of the whole engine is related to only a few important parameters such as the thrust required at sea level. The model is based on historical data and surrogate models. For example, D. Raymer [354] provides the weight and the TSFC of a turbofan as a function of the thrust required, the maximum Mach number, and the bypass ratio. C. Svoboda [424] suggests an equation that relates the key engine parameters (weight, dimensions, TSFC, etc.) to the thrust required. The methods from this approach are very fast to run, easy to implement, and require only few and usually available inputs. Nevertheless, their accuracy may be relatively low, with an error generally greater than 10%. Moreover, this approach does not enable the comparison between engines with different cycles or technology levels. Periklis Lolis [262] compared the different methods for engines with a weight up to 9,000 kg. He showed that Raymer's method leads to 240% errors while Svoboda's method leads to less than 30%.

- Component-based approach: in this approach, the weight of the engine is computed by summing the individual weight of each component such as the compressor, the inlet, the combustion chamber, etc. For example, Pera et al. [333] developed equations for V/STOL applications. One of the most generic tools, widely known and used, is the NASA's Weight Analysis of Turbine Engine (WATE) software. It benefits from a very good accuracy (errors usually between 5% and 10%) but requires many design variables (as do the other tools in this category). It also allows users to improve or extend its capabilities in order to match their specific needs. Tools based on a computer-based approach compute the weight and the characteristics of each component before integrating them in order to find the overall engine performance: weight, dimensions, TSFC, etc. The computation time is often significant and the correlation factors are becoming outdated and thus need to be updated in order to ensure a good

accuracy.

- Cycle parameters-based approach: this approach uses the main cycle parameters such as the mass flow rate $\dot{m}$, the thrust required at sea level ($T_{SL}$) the Turbine Inlet Temperature (TIT), the Overall Pressure Ratio (OPR), and the Bypass Ratio (BPR). This approach is a good compromise between the accuracy of the component-based approach and the simplicity (few inputs) of the single parameter approach. It uses both historical data and correlation factors in order to match the engine weight. Programs such as PARA, PERF, and AEDSys provide the main performance parameters as a function of the flight conditions, the mass rate, the pressure and bypass ratios, the maximum allowable temperatures and components efficiencies. E. Torenbeek [442] provides Equation 29, which enables the weight estimation without any spatial dimensions.

$$W_{eng} = \frac{10 \times \mathrm{OPR}^{0.25} \times \dot{m}}{1 + \mathrm{BPR}} + 0.12 \frac{T_{SL}}{g_0} \left( 1 - \frac{1}{\sqrt{1 + 0.75\mathrm{BPR}}} \right) \tag{29}$$

  Gerend and Roundhill [169] provide a series of formulas that relate the different design parameters to the engine weight. These formulas, developed in 1970, have been updated by R. Quintero as part of his PhD thesis [351]. In addition to the main cycle parameters, they also use parameters that include the certification year of the engine or its forecast lifetime.

The evaluation and the comparison of these tools are provided in Table 17 leveraging past studies such as the ones from P. Lolis [262] and B. Montgomerie [298].

**Rocket engines:** Similarly, the same three approaches can be defined for rocket engines. Component-based tools are based on Navier-Stokes equations and CFD codes in order to compute the flow phenomena within each component of the engine. While being very accurate, these tools are very slow to run and require a huge amount of time for component modeling and meshing. Cryogenic Rocket Combustion (CryoROC) is a software developed by Astrium to characterize complex flows in the combustion chambers and nozzles of

**Table 17:** Comparison of the various weight estimation tools

| | Few inputs and fast | Acceptable accuracy | Architecture comparison | Weight | Performance |
|---|---|---|---|---|---|
| **Svoboda** | ✓ | ✓ | | ✓ | ✓ |
| **Raymer** | ✓ | ✓ | | ✓ | ✓ |
| **WATE** | | | ✓ | ✓ | ✓ |
| **Gerend-Roundhill** | ✓ | ✓ | ✓ | ✓ | |
| **Torenbeek** | ✓ | | ✓ | ✓ | ✓ |
| **AEDSys** | ✓ | ✓ | ✓ | | ✓ |
| **PARA or PERF** | ✓ | ✓ | ✓ | | ✓ |

cryogenic hydrogen/oxygen rocket engines [179]. Rocket Combustion Flow Analysis Module (ROCFLAM) is also a multi-phase Navier-Stokes code for flow phenomena in thrust chambers when storable propellant combinations such as hydrazine/$N_2O_4$ and $H_2/N_2$ are used [249]. Many software using the cycle parameters-based approach already exist for performance analysis and are very similar: Rocket Propulsion Analysis (RPA), Cpropep, Redtop, etc. These tools take the main design parameters such as the propellants, the pressure of the combustion chamber, and the nozzle area ratio and provide the optimal mixture ratio, the exit speed, the thrust coefficients, and the specific impulse as a function of altitude. Redtop is also able to handle weight estimation but instead of outputting a single value, it provides a confidence interval to account for users not selecting important parameters such as the material. Instead, R. Humble [216] provides analytic formulas based on historical data for each type of rocket engine in order to compute the weight and the size of each component. The single parameter approach uses the typical ideal rocket equations as well as historical data. In addition, R. Humble [216] also provides Equation 30 to get a first estimate of liquid engine mass $m_e$ based on its required thrust $T$.

$$m_e = \frac{T}{g_0 \left(25.2 \log T - 80.7\right)} \tag{30}$$

#### *3.3.3.4 Tool Selection*

While cycle parameters-based approach can be used to estimate the weight and size of jet engines, there is no available tool that meets all requirements for rocket engines. Indeed, since jet engines are widely used and well-known, historical data can be used to predict both weight and performance. As far as rocket engines are concerned, a physics-based modeling is required for both weight and performance calculations. Hence, a hybrid method will be developed. Firstly, surrogate models of basic engine performance parameters will be developed using the cycle-based software RPA. Secondly, using these parameters, empirical and physics-based relations from Humble will be used to predict the weight and size of all rocket engines. Finally, surrogate models will be calibrated based on existing engines using efficiency factors. These observations lead to the following Hypothesis:

**HYPOTHESIS 3.1: IF performance parameters found by creating surrogate models of the cycle-based software Rocket Propulsion Analysis (RPA) are inputted into a physics-based weight prediction model THEN performance and weight of liquid, solid, and hybrid rocket engines can be rapidly predicted at a conceptual design level.**

In order to validate this Hypothesis, Experiment 3.1 is implemented. As inputs, it requires a description of existing engines and their requirements found in literature. The experiment consists in computing weight and performance of these engines and check the results. To fully validate Hypothesis 3.1, two elements need to be verified: the accuracy of the evaluation method through a comparison with actual data for both performance and weight/size, and the ability of the method to be used at a conceptual design level. This will be done through four questions:

1. Is the performance of all chemical rocket engines obtained similar to real data?
2. Are the weight and size of all chemical rocket engines obtained similar to real data?
3. Are the engines described by no more than 4-5 variables?

4. Is the calculation fast (around 1 second) so that it can be integrated into a large-scale optimization process?

To describe the behavior of a flying vehicle, the four external forces acting on this vehicle must be modeled: weight, thrust, lift, and drag. While Sections 3.3.2 and 3.3.3 enable the characterization of the first two, the following section will discuss the two aerodynamic components: lift and drag.

### 3.3.4 Aerodynamic Modeling

Aerodynamic modeling is one of the cornerstones of a new vehicle design. Indeed, to compute the trajectory and the vehicle performance, the main aerodynamic coefficients must be determined. This aerodynamic study is a very difficult task since suborbital vehicles go through all flight regimes presented below:

- $M \in [0, 0.8]$: The subsonic regime corresponds to Mach numbers that do not involve shocks and supersonic regions along the vehicle. These flows are characterized by smooth streamlines as well as a flow propagation both downstream and upstream. For Mach numbers smaller than 0.3, the flow can be considered as incompressible (constant density).

- $M \in [0.8, 1.2]$: The transonic regime involves Mach numbers close to one and is a hybrid domain that includes both subsonic and supersonic regions along the vehicle. It is usually characterized by high drag coefficients and consequently requires a significant amount of thrust.

- $M \in [1.2, 5]$: The supersonic regime is defined by a flow with a Mach number greater than one at every point. These flows are usually characterized by shock and expansion waves. One of the most important properties of supersonic flow is that disturbances cannot propagate upstream.

- $M \geq 5$: The hypersonic regime corresponds to very high supersonic speeds and is mainly characterized by high temperature effects and interactions between shock waves and viscous boundary layers.

For each regime, the behavior of the flow is very different and implies different assumptions. Therefore, many aerodynamic tools are only valid for certain regimes. Existing concepts reach Mach numbers around 4-5 so that the aerodynamic module must handle low speed calculations for take-off and landing purposes as well as hypersonic phases.

In addition to the flight velocity, another key driver of the aerodynamic parameters is the shape of the vehicle. As presented in Section 1.2, suborbital concepts do not have a standard geometry: winged bodies (delta wing, swept wing,. etc.), slender bodies as well as other unconventional configurations such as the rotating wing of the SpaceShipTwo.

By definition, the vehicle must be able to cross all atmospheric layers, from the ground to the space boundary. Therefore, it will be exposed to various atmospheric conditions in terms of density and temperature that will greatly impact its aerodynamic behavior and that must be taken into account.

Thus, the aerodynamic module must be able to handle all these configurations under all flight regimes. Moreover, technical requirements previously mentioned must also be met: fast to run, automation, availability, conceptual design level, and easy to use. This section first describes the atmosphere which is the operating environment of the vehicle. Then, possible configurations that must be modeled will be presented and detailed. An overview of the different aerodynamic analysis approaches will be provided and existing tools compared and evaluated. The ultimate purpose of this section is to find a tool, or a set of tools, that can constitute the aerodynamic module of the integrated design environment.

#### *3.3.4.1 Evolution of the Main Atmospheric Parameters*

The International Organization for Standardization (ISO) published a standard atmospheric model as an international standard: ISO 2533:1975. It defines the International Standard Atmosphere (ISA) with a set of tables that describe the key atmospheric parameters: pressure, temperature, density, etc. The temperature behavior also suggests a decomposition of the atmosphere into several layers, as displayed in Figure 37.

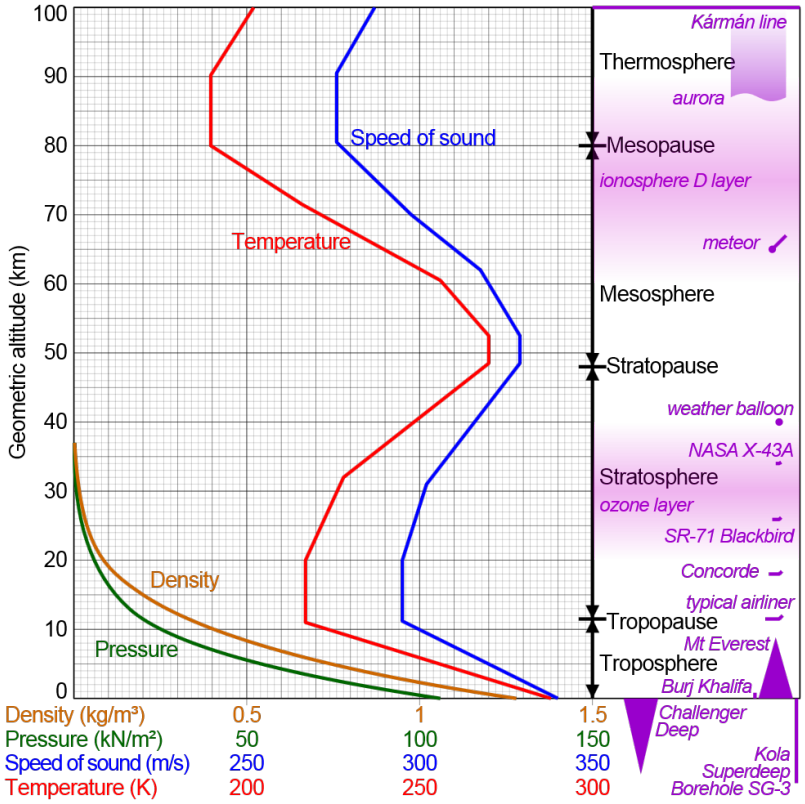



**Figure 37:** Evolution of the key atmospheric parameters [89]

Several mathematical models of the parameters are available. The temperature is usually modeled by a piece-wise linear function presented in Equation 31, where $T$ is the temperature, $h$ the altitude, and the set $(T_1, h_1, a)$ three constants specific to the model and the layer concerned [43].

$$T(h) = T_1 + a\,(h - h_1) \tag{31}$$

Similarly, the density is modeled by an exponential or a power law such, as presented in Equation 32, where $T_0$ and $\rho_0$ are respectively the standard temperature and density at sea level and $T_h$ another constant [43].

$$\rho(h) = \rho_0 \left(1 + \frac{T_h}{T_0} h\right)^{4.26} \tag{32}$$

Finally, complete models developed from tables and more precise interpolations exist such as the one developed by B. Lewis into Matlab [256]. Thus, they can directly be used in the integrated environment when necessary.

#### *3.3.4.2 Description of the Possible Vehicle Configurations*

All primary airframe architectures that can be used for suborbital vehicles are assumed to fit into one of the two following categories: winged-body and slender-body vehicles.

A slender body is the simplest shape since it is only composed of a large cylinder closed by a cone at the front. Small ailerons can be included at the rear of the fuselage in order to control and stabilize the vehicle. The main design variables of such vehicles are the cross-section area, the radius of the cone, the relative length of the cone compared to the entire body, etc.

A winged body can be aerodynamically decomposed into three main components: the fuselage, the wing, and the empennage (nacelles could also be included if needed). Each of them has specific roles: the fuselage carries the payload, the rocket engine, a part of the propellant, and other subsystems. The wing creates lift and is equipped with ailerons to control the roll motion. The empennage is mainly used for pitch and yaw control and stability. The fuselage can be described similarly to the slender body. The wing must be defined by specific parameters such as the sweep angle, the taper ratio, etc. As far as the empennage is concerned, different configurations can be selected depending on the type of wing.

These shapes can be aerodynamically defined by a series of coefficients that are used to characterize the vehicle's behavior throughout its flight regimes. Among these coefficients, there are the drag polar coefficients and the maximum lift coefficient. To model them, an aerodynamic analysis must be performed and the possible approaches are described below.

#### *3.3.4.3 Analysis of the Main Aerodynamic Analysis Approaches*

Aerodynamics is based on the study of the fundamental Navier-Stokes equations. They describe the motion of fluid substances and include the continuity equation (33a), the

momentum equation (33b), and the energy equation (33c).

$$\frac{\partial}{\partial t}\iiint_{\mathcal{V}} \rho \, \mathrm{d}\mathcal{V} + \oiint_{S} \rho \mathbf{V} \cdot \mathbf{dS} = 0 \tag{33a}$$

$$\frac{\partial}{\partial t}\iiint_{\mathcal{V}} \rho \mathbf{V} \, \mathrm{d}\mathcal{V} + \oiint_{S} (\rho \mathbf{V} \cdot \mathbf{dS}) \, \mathbf{V} = -\oiint_{S} p \mathbf{dS} + \iiint_{\mathcal{V}} \rho \mathbf{f} \, \mathrm{d}\mathcal{V} + \mathbf{F_{visc}} \tag{33b}$$

$$\begin{aligned}\frac{\partial}{\partial t}\iiint_{\mathcal{V}} \rho \left(e + \frac{V^2}{2}\right) \mathrm{d}\mathcal{V} + \oiint_{S} \rho \left(e + \frac{V^2}{2}\right) \mathbf{V} \cdot \mathbf{dS} = \iiint_{\mathcal{V}} \dot{q} \rho \, \mathrm{d}\mathcal{V} - \oiint_{S} p \mathbf{V} \cdot \mathbf{dS} + \\ \iiint_{\mathcal{V}} \rho \left(\mathbf{f} \cdot \mathbf{V}\right) \mathrm{d}\mathcal{V} + \dot{\mathbf{Q}}_{\mathbf{visc}} + \dot{\mathbf{W}}_{\mathbf{visc}}\end{aligned} \tag{33c}$$

The different approaches that exist to perform the aerodynamic analysis come from the different assumptions that are made to simplify the complex equations previously mentioned.

**Datasheet Methods:** Due to their complexity, the Navier-Stokes equations are difficult to solve. Therefore, one solution is to create a database of aerodynamic coefficients that come from experimental measurements. This method is particularly useful for performance assessment of conventional concepts but does not allow any shape optimization or aerodynamic trade-off study. For example, estimations of drag polar coefficients and maximum lift coefficient are fixed at a given value for each type of aircraft [271].

**Analytical Methods:** Under some assumptions, analytical formulas can be derived to predict the aerodynamic coefficients. The lifting-line theory is one of the most famous examples of such analyses. While being extremely fast, easy to implement, and parametric, this theory faces several limitations. Indeed, it cannot be used for compressible flows, viscous flows, swept wings, and low aspect ratio wings. To overcome some of these limitations, correction factors have been included in analytical formulas. For example, the Prandtl-Glauert factor $\beta$ presented in Equation 34 takes into account compressibility effects, where $M_\infty$ is the free stream Mach number.

$$\beta = \sqrt{1 - M_\infty^2} \tag{34}$$

Another way to obtain analytical formulas is to build empirical or semi-empirical relations based on experimental measurements, observations, and correction factors.

Analytical formulas often rely on restrictive assumptions and application domains. Nevertheless, they are often used in the early design phases because they do not require long computational time and are usually parametric. The large number of available formulas provides a good coverage of typical configurations with more or less accuracy and level of detail.

**Computational Fluid Dynamics Methods:** CFD methods are more complex than the other two because they require meshes and numerical resolutions but allow designers to overcome some of the limitations previously exposed. Indeed, CFD can handle complex configurations and intake flows with multi-element design. Moreover, it allows detailed trade-offs and can be combined with optimization tools. However, CFD methods require more computational time and work from designers to mesh and define the shape so that they are considered to be more expensive than the others.

There exists a large variety of CFD methods currently available. Figure 38 presents a typical decomposition going from the most general (top) to the most specific (bottom) [103].

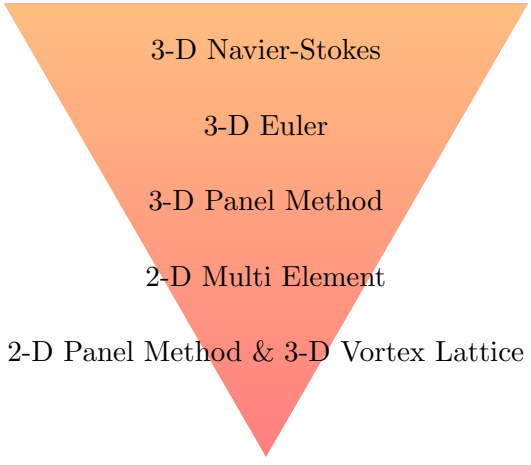



**Figure 38:** Various CFD methods

A 3-D resolution of the Navier-Stokes equations can theoretically handle every problem

but appears to be extremely expensive in terms of time and computer resources. It also requires an extremely accurate meshing of the studied body to be usable. Therefore, a review of the assumptions under each method is necessary in order to choose the most suitable method. Nevertheless, the study will be restricted to 3-D methods.

3-D Navier-Stokes equations only assume a continuum fluid and are always valid for atmospheric applications. One of the most famous implementations of such resolution is the product suite ANSYS Fluid Dynamics [15]. In particular, ANSYS gathered ANSYS CFX and ANSYS Fluent into the single ANSYS CFD bundle. This tool has a wide variety of applications such as automotive, aerospace, and maritime industries. Once the body has been meshed, this software is able to capture all flow phenomena: laminar and turbulent flows with transition, incompressible and compressible flows, all flight regimes, heat exchanges, static or dynamic bodies, etc. Thus, this tool allows designers to fully characterize every vehicle under all flight conditions. Nevertheless, it requires a precise and non-parametric meshing of the surface as well as a very long computational time and heavy CPU requirements. This software is usually used for a detailed sizing and optimization of pieces such as fairings, pylons, rotor blades, etc.

Euler equations directly come from the Navier-Stokes equations with certain assumptions. Indeed, the flow is considered to be inviscid: no viscosity, no heat conduction, and no mass diffusion. This approximation is valid if the Reynolds number is high, which is the case in the context of this research. By removing some negligible terms, this assumption will speed up the calculation process while keeping a reasonable accuracy. The implementation of this method is similar to the previous one and some of the most popular and well-known software are MGAERO [416] and Flite3D [436]. Nevertheless, this method still requires an important CPU time and a precise meshing.

Panel methods are based on potential flow assumptions defined by an irrotational flow. The surface is discretized into small panels upon which singularities are attached. Application of boundary conditions allows to solve Laplace's Equation. This assumption greatly simplifies the problem but is only valid for subcritical flows. Initially developed for incompressible flows, this method has been extended to compressible flow and remains roughly

accurate for supercritical and supersonic flows with attached shocks. It becomes inaccurate for detached shocks at high supersonic speeds. Implementations of this method is considerably faster than the resolution of the Euler equations and the meshing does not need to be as precise as for the two previous methods. VSAERO is one implementation of this method and examples of complete detailed vehicle resolution show a computational time of several hours and is still too long for a conceptual design space exploration [103, 417]. However, its application on simple surfaces such as simple wing and fuselage remains relatively fast to run and usable.

The 3-D vortex lattice method is a method mainly used in early design phases. This method assumes that the flow is incompressible, incviscid, and irrotational. Nevertheless, compressibility effects at subsonic speeds can be taken into account using correction factors such as the one presented in Equation 34. Another limitation of this method is its restriction to thin surfaces so that slender bodies cannot be modeled. Moreover, the angle of attack is also assumed to be small. This method can be seen as an extension of the lifting-line theory by creating an infinite number of horseshoe vortices in order to model the lifting surface. This method has been widely implemented in different languages to create multiple tools. Among them, there are AVL [122], Tornado [356], and VLM [490].

Based on the previous comparison, panel methods and 3-D vortex lattice methods seem to be the best trade-off between fast computational time, accuracy, and ability to handle all configurations and flight regimes. Nevertheless, at a conceptual design level and for design space exploration purposes, the use of empirical formulas would highly speed up the overall optimization process. Hence, the following comparison will only consider extremely fast computational methods and empirical formulas.

#### *3.3.4.4 Comparison of Existing Tools*

The complexity of the aerodynamic analysis does not allow the selection of a single tool fast enough to perform an analysis at the vehicle level over the entire flight regime and for every configuration. Hence, the drag will be decomposed into different types so that simpler and faster tools or methods can be independently used for each category with a relatively

good accuracy. The different categories of drag considered in this study are listed below and represented by their corresponding drag coefficient:

- Friction and form drag: $C_{Dff}$
- Induced drag: $C_{Di}$
- Interference drag: $C_{Dif}$
- Wave drag: $C_{Dw}$

Therefore, the entire vehicle drag can be found be summing each component as described in Equation 35. Because of the differences in the nature of each phenomenon, different methods will be used for each of them in order to maintain good accuracy without using complicated and long CFD calculations.

$$C_D = C_{Dff} + C_{Di} + C_{Dif} + C_{Dw} \tag{35}$$

**Skin friction and form drag:** The form drag is a direct consequence of the shape of the body. Indeed, by perturbing the flow field around it, the body creates a net imbalance of surface pressure acting in the drag direction. The integration of this pressure imbalance over the surface directly results in the form drag. The latter is mainly affected by the body shape, the angle of attack, and shocks. Skin friction is caused by the net effect of shear stress acting in the drag direction. This friction drag is affected by the smoothness of the surface and the size of the wetted area.

In the literature, the estimation of these drags is mainly dominated by two main methods: the equivalent skin-friction method and the component buildup method [354]. The first method assumes that the form drag is a small percentage of the skin friction so that Equation 36 can be applied. $C_{fe}$, called the equivalent skin friction coefficient, depends on the type of aircraft.

$$C_{Dff} = C_{fe}\frac{S_{wet}}{S_{ref}} \tag{36}$$

The component buildup method is more precise and estimates the drag coefficient of each component independently. Contrary to the equivalent skin-friction method, this method

can enable a geometry optimization and trade-off studies. The skin friction and form drag can be estimated in Equation 37. It uses the flat-plane skin friction coefficient $C_{f_k}$ corrected by a form factor $FF_k$.

$$C_{Dff} = \sum_k C_{f_k} FF_k \frac{S_{wet_k}}{S_{ref}} \tag{37}$$

Different models of the form factor are available in the literature and are compared by Gur et al. [187]: Torenbeek [442], Jobe [222], Nicolai [319], Raymer [354], Shevell [395], and Hoerner [205].

Roskam presents a specific methodology that also incorporates the interference drag [369]. This method appears to be highly parametric but relies on parameters from experimental data provided by various plots. These plots need to be mathematically modeled in order to automate this method and make it usable.

**Induced drag:** Every finite lifting body, and especially wing, induces wing-tip vortices. These vortices result in an additional pressure drag component called induced drag. Gur et al. [187] cover several numerical estimation models usually used: the Trefftz plane [37], Prandtl's lifting-line theory [338], the vortex lattice method [138], and the Weissinger non-linear lifting-line model [475]. Simpler methods consist in assuming that the induced drag takes the form presented in Equation 38, where $C_L$ is the lift coefficient and $AR$ the aspect ratio. Then, several empirical models are available to find the efficiency parameter $e$ such as Raymer [354], Roskam [369], Sadraey [375], etc.

$$C_{D_i} = \frac{C_L^2}{\pi e \mathrm{AR}} \tag{38}$$

**Interference drag:** It has been shown that the total drag of combined bodies is greater that the sum of its components' drag [21]. Hence, the interference drag is defined as a pressure drag due to the mixing of flow fields around each component.

Hoerner developed a series of empirical relationships for various combinations of configurations using experimental data [205]. However, his results are restricted to subsonic flow fields. Tétrault extended Hoerner's formulation to the transonic regime for a wing-fuselage application [432]. He used CFD calculations to develop his formula whose application is

restricted to wings with thickness-to-chord ratios smaller than 7.5%. According to his formulation, the wing-fuselage interference drag depends on the thickness-to-chord ratio, the Reynolds number, and the inclination angle (angle between the wing and the surface's normal). Raymer uses an interference factor which depends on the interaction [354]. This factor increases the parasite drag of each element in order to take into account the interference drag.

**Wave drag:** At supersonic speeds, flow features are dominated by shock waves. These shock waves are responsible for a pressure pattern characterized by a strong pressure imbalance in the drag direction. Hence, the integration of this pressure difference over the entire body defines the wave drag, which is a particular type of pressure drag. Its corresponding drag coefficient is usually decomposed into two components: the zero-lift wave drag $C_{Dw,0}$ and the wave drag due to lift $C_{Dw,l}$, as stated in Equation 39.

$$C_{Dw} = C_{Dw,0} + C_{Dw,l} \tag{39}$$

Preliminary analyses of supersonic drag usually rely on linearized or modified linearized methods. Such formulations have been developed by Loxam [263], Jones [225], Jumper [226], etc. One of the most widely used applications is the Harris wave drag program called AWAVE [195, 196], a program for zero-lift wave drag calculation. While this tool can only handle specific types of configurations, Rallabhandi et al. [352] developed an improved wave drag code that requires a complete meshing of the vehicle. NASA Ames Research Center developed the Arrow code that can compute the wave drag due to lift of arrow wings at supersonic speeds based on Rogers' theoretical developments [363]. Roskam decomposes the vehicle into components and presents a methodology for wave-drag calculation for each of them based on empirical relationships and coefficients that can be extracted from experimental results [369]. Finally, Raymer proposes a modified version of the skin friction drag estimation including adjustment factors [354]. Special attention is given to the fuselage wave drag because of the large impact of its shape on wave drag. To do so, the perfect Sears-Hack body is usually used to assess the zero-lift wave drag coefficient [421].

The previous analysis shows that Roskam provides a methodology that meets all required capabilities: empirical relations modeling all drag coefficients for all geometries and all flight regimes. By relying on a single tool, results will also be more consistent so that comparison inconsistencies will be avoided. Up to now, Sections 3.3.2 to 3.3.4 aimed at characterizing vehicles' performance and weight as individual disciplines. These different disciplines must now be gathered to model the overall trajectory. This is addressed in the following section.

### 3.3.5 Trajectory Optimization

In order to study the trajectory of an airborne vehicle, its governing equations of motion through the air must be established and understood. Figure 39 represents the four physical forces acting on an aircraft once airborne. The free-stream velocity $V_\infty$ is defined as being parallel to the flight path which makes an angle $\theta$, called flight path angle, with respect to the horizontal. The weight $W$ is directed towards the center of the Earth which is assumed to be confounded with the vertical. The resultant of the aerodynamic forces is traditionally separated into two forces projected onto two different axes: the lift $L$ which is perpendicular to the flight path and the drag $D$ which is parallel to the flight path. Finally, the thrust $T$ is defined forward to the vehicle inclined with an angle $\epsilon$ with respect to the flight path.

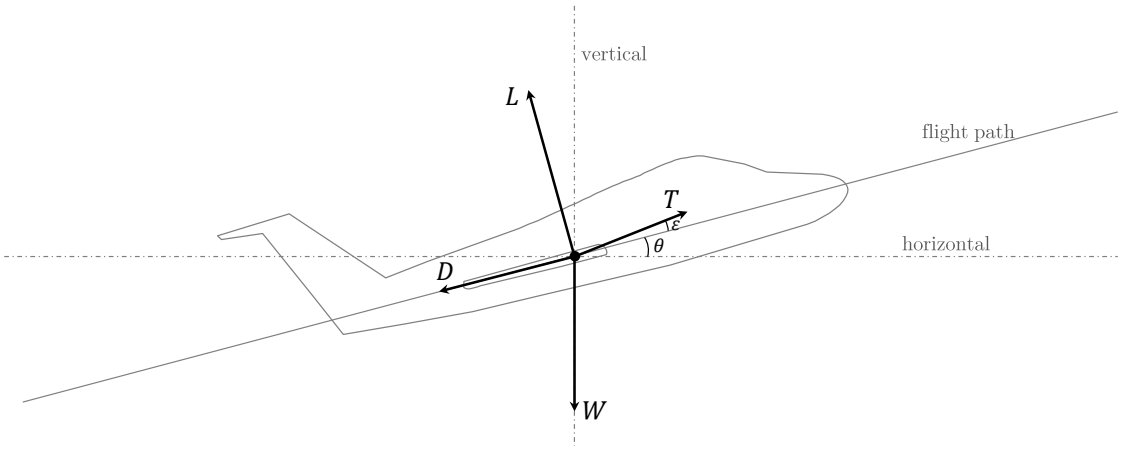



**Figure 39:** Forces acting on an airborne vehicle

In addition to these notations, the angle between the lift and the Earth's vertical plane containing the aircraft's longitudinal axis vertical is called the bank angle $\phi$. The instantaneous curvature of the flight path angle in the horizontal plane is defined by $R_1$ and the

one in the vertical plane is defined by $R_2$. In this study, the Earth is assumed to be flat and its reference frame to be an inertial one. Moreover, the vehicle is modeled by a point mass (rotational motions neglected) and no slip angle is considered. Thus, the vehicle's motion can be described using Equations 40, 41, and 42 [13].

$$m\frac{dV_\infty}{dt} = T\cos\epsilon - D - W\sin\theta \tag{40}$$

$$m\frac{V_\infty^2}{R_1} = L\cos\phi + T\sin\epsilon\cos\phi - W\cos\theta \tag{41}$$

$$m\frac{(V_\infty\cos\theta)^2}{R_2} = L\sin\phi + T\sin\epsilon\sin\phi \tag{42}$$

A first natural approach to study and optimize aerospace trajectories would be to find a direct solution to these equations. Even with typical approximations, this complete resolution is usually complex and requires highly sophisticated mathematical algorithms that are time consuming. A second, simplified, approach has been developed and uses the vehicle's total energy in lieu of these equations.

#### *3.3.5.1 Complete Resolution of the Equations of Motion*

This first approach directly uses the equations of motion and the natural state variables such as position, velocity, and time. Assuming that the lateral motion is irrelevant (good approximation at conceptual design phase) for the trajectory calculation, a two dimensional formulation can be used [296, 455]. It consists in a system of six equations and six unknowns $(x, y, u, v, t, \gamma)$ where $x$ and $y$ are the coordinates and $u$, $v$ the velocity components on the x-axis and y-axis respectively. $t$ is the time and $\gamma$ the thrust vector angle defined by the sum of $\theta$ and $\epsilon$. Equations 43 to 48 present the six equations of the system.

$$\dot{x} = u \tag{43}$$

$$\dot{y} = v \tag{44}$$

$$\dot{m}\ddot{x} = T\cos\gamma - L\sin\theta - D\cos\theta \tag{45}$$

$$\dot{m}\ddot{y} = T\sin\gamma + L\cos\theta - D\sin\theta \tag{46}$$

$$\dot{m} = -\frac{T}{g_0 Isp} \tag{47}$$

$$\gamma = \gamma(t) \tag{48}$$

In addition to this system of equations, a series of constraints might be imposed to the problem. These constraints come from requirements (maximum load factor), structural limitations (maximum dynamic pressure), etc. Hence, the problem becomes a non-linear constrained optimization problem. This is only an example of simplifications that can be done on the original system. Some complex formulations even consider the lateral motions and rotations [33, 35, 114]. To solve the aforementioned system or even a more complete formulation, several mathematical methods have been developed and are detailed below.

Single shooting methods are numerical methods for solving a boundary value problem by reducing it to the solution of an initial value problem and propagating the solution throughout the trajectory. Two types of implementation have emerged: direct and indirect. The main difference between the two implementations relies on the definition of the control function. For direct shooting methods, the control is parameterized and an explicit numerical integration is performed. For indirect methods, the control is defined at each step by the maximum principle. Shooting methods have been implemented in several software such as the Generalized Trajectory Simulation (GTS) [290], DUKSUP [26] and the Program to Optimize Simulated Trajectories (POST) [48]. Written by Lockheed Martin Astronautics and NASA-Langley Research Center (LaRC), the latter is widely used in the aerospace domain. It can handle problems with both three and six degrees of freedom. Initially developed to optimize the trajectory of the American Space Shuttle, it is capable of dealing with powered and unpowered vehicles for atmospheric ascent and re-entry missions. It is

based on a direct shooting approach and, by construction of this approach, the solution is always physically correct since there is no risk of any discontinuity in state variables. Aerodynamic coefficients are inputted by using lift and drag models or axial and normal models. Propulsion data can be inputted for three types of engines: rocket engines, jet engines, and ramjet engines. This approach is especially suitable when the problem can be described by a small number of variables. However, a major issue persists for all these methods: small changes introduced early in the trajectory could propagate into highly non-linear changes at the end of the trajectory and result in non-accurate solutions. To address this issue, a simultaneous approach has been introduced with the multiple shooting approach.

To overcome the propagation problem inherent to sequential resolutions, the multiple shooting approach breaks the trajectory into smaller intervals and integrates the different equations on each one of these intervals. In order to ensure the overall accuracy, it imposes additional matching constraints at the boundary between two segments. Similar to the single shooting approach, both direct and indirect algorithms can be implemented to compute the values of the state variables as a function of time on the whole interval. This approach greatly improves the robustness of the calculation but requires more variables and the size of the problem rapidly increases with the number of intervals. Nevertheless, this drawback is mitigated by the ability of this approach to exploit parallel processing techniques. This approach has also been widely implemented within the aerospace industry in tools such as BNDSCO [322, 422] and Kreim et al. [243]. They have been respectively applied to study missile trajectories and the re-entry phase of the American Space Shuttle's trajectory.

The collocation approach, which is very similar to the multiple shooting approach, also divides the trajectory into small segments. However, in addition to discretizing control variables, this approach also discretizes state variables. To be optimized, the trajectory is divided into phases also called sub-arcs, linked by events. Different types of phases can be defined that are divided into two groups:

- Propagation phases: analytic propagation, explicit integration, implicit integration, etc.

- Transformation phases: jump phase (state relation discontinuities), state transformation (changes in equations of motion), transformation of coordinates, etc.

Phases from the second category are discrete and are used to link phases from the first category that last a finite amount of time. This approach still has a drawback since there is a possible emergence of non-physical solutions due to discontinuities in state variables. Indeed, inaccurate connections between two partitions of a state variable would create non-physical trajectories with jumps in variables such as altitude or speed. This approach is still considered as very efficient for trajectory optimization problems and was implemented in various aerospace tools: Optimal Trajectories by Implicit Simulation (OTIS) [193], Sparse Optimal Control Software (SOCS) [433], Advanced Launcher Trajectory Optimization Software (ALTOS) [64], etc. OTIS was written by the Boeing Corporation in collaboration with NASA-Glenn Research Center (GRC) and is one of the most widely used in the U.S. industry with POST. It performs trajectory performance studies at a relatively high-fidelity level [309, 315, 342, 358]. Written in Fortran, this program is able to handle a large panel of objects (aircraft, missiles, re-entry vehicles, satellites, etc.) with multiple types of propulsion systems such as air-breathing and rocket engines. The aerodynamic and propulsion models used to compute the trajectory must be defined by the users. Then, for a given vehicle, OTIS determines the best trajectory such that the vehicle reaches an inputted altitude while meeting a series of specified constraints: maximum heat flux, stall angle, etc.

D. Nelson [315] compares the two U.S. software using test cases and concludes that, even if the two approaches give the same quantitative results, some differences have to be noticed:

- OTIS allows the users to easily plot all variables while POST requires more work from the users.

- Users can easily create their own variables in OTIS. However, the creation of new variables in POST requires major changes in the source code.

- POST requires an initial value for each variable. Thus, an unwise choice of these guessed values could result in non-optimal solutions. In contrast, OTIS only requires

the users to specify the variables that they want to track and does not need initial values for these variables.

- Contrary to POST, OTIS' implicit method could output physical inaccuracy in state variables such as discontinuity.
- POST has the capability to impose a maximum load factor while the latter requires a lot of efforts in OTIS. This capability provides a good advantage for manned vehicles.
- Contrary to POST, OTIS does not allow the users to model horizontal take-off phases. In order to overcome this issue, constraints can be added but they deteriorate the accuracy of the results.
- Some small issues have also been found while modeling air-breathing engines in OTIS.

In a nutshell, the previous methods are based on a complete, but complex, resolution of the equations of motion. Therefore, when a detailed description of the vehicle is available and the methods well implemented, the results are very accurate. Nevertheless, the computation time of those methods is relatively long for a design space exploration and the data available at a conceptual design level might be insufficient for a successful use of these methods. Hence, a more simplified formulation is desirable for our requirements, as described in the next section.

#### *3.3.5.2 Energy-State Approximation*

While the previous approach provides a complete numerical resolution of the two or three equations of motion, there is a need to develop a simplified model. Indeed, complex models only yield to more accurate results if they closely represent the real system. However, at a conceptual design level, detailed information about the system is usually unavailable. In addition, a complete resolution might require a long computation time and a faster approach becomes necessary to explore the entire design space. To address these challenges, the ESA method has been introduced by E. S. Rutowski [373] and developed by Bryson et al. [61]. This method assumes that both $\epsilon$ and $\alpha$ are small angles and defines the total energy per unit mass $E_s$ as a new state variable. It includes the contribution of both the potential

energy and the kinetic energy and can be defined using Equation 49, where $h$ represents the altitude, $V_\infty$ the free stream velocity, and $g_0$ the acceleration of gravity.

$$E_s = g_0 h + \frac{V_\infty^2}{2} \tag{49}$$

With the aforementioned assumptions, the time rate of change of the total energy $\dot{E}_s$ can be expressed using Equation 50, where $T$ is the thrust, $D$ the total drag, and $m$ the vehicle mass.

$$\dot{E}_s = V_\infty \frac{T - D}{m} \tag{50}$$

It can be shown that $h$ and $\alpha$ can be expressed as a function of $E_s$ and $V_\infty$ only. In this case, $V_\infty$ appears as a control variable. For a given $E_s$, the minimum time-to-climb trajectory can be formulated using Equation 51, while the minimum fuel-to-climb trajectory can be formulated using Equation 52.

$$\max_{V_\infty} \left\{ \left[ T\left(E, V_\infty\right) - D\left(E, V_\infty\right) \right] V_\infty \right\} \tag{51}$$

$$\max_{V_\infty} \left\{ \frac{T\left(E, V_\infty\right) - D\left(E, V_\infty\right)}{\dot{m}} V_\infty \right\} \tag{52}$$

Initially applied to aircraft trajectory optimization problems, it has been shown that this approach is also suitable for launchers and suborbital vehicles [200]. This ESA method benefits from a faster execution time, a level of detail corresponding to the stated problem and a reduced complexity or risk of failure compared to the complete resolution. Therefore, this approach seems to be more appropriate although it is less accurate. Many tools have already successfully implemented this method for similar analyses [60, 214, 339, 455].

Sections 3.3.2 to 3.3.5 bring the required pieces to assess the flying performance of the vehicle. However, the design and engineering of a product are inseparable from the consideration of its entire life-cycle costs. Hence, the following section discusses the different life-cycle cost components.

### 3.3.6 Life-Cycle Cost Assessment

In today's highly competitive environment, life-cycle cost estimation is becoming a significant discipline for new business endeavors. Indeed, good and relatively accurate estimation techniques are required, even in early design phases, in order to avoid huge investment of time and money in non-viable products. However, due to the inherent iterative and evolving nature of the design process, cost estimation appears to be a very difficult task, especially at the early stages. The small amount of data available about the concept brings even more uncertainty and risk. Besides, the classified nature of some projects in the very competitive aerospace industry makes the collection of data tedious and difficult. Nevertheless, the latter is essential for both assessing the economic viability of the project and making it as affordable as possible [66]. To address these challenges, this section defines the basic components of life-cycle costs and provides a comparative overview of the current and potentially applicable state-of-the art cost estimation techniques.

#### *3.3.6.1 Life-Cycle Cost Components*

To support the establishment of a new suborbital market, the economic viability of the vehicle manufacturer and operator must be ensured. On the one hand, the manufacturer deals with Research, Development, Testing and Evaluation (RDT&E) costs and production costs. Its revenues come from the money earned from selling the vehicle to the operator. Consequently, the manufacturer's Return On Investment (ROI) is optimized by maximizing the difference between the vehicle price and the aforementioned costs. On the other hand, the operator deals with the acquisition cost, the total operating cost and the disposal cost of the vehicle. Its revenues come from the passengers and depend on the ticket price. Its ROI is maximized by reducing acquisition and total operating costs while maximizing the ticket price and the number of passengers. While the two analyses seem independent, they are in fact highly correlated. In addition to the impact of the acquisition price on both manufacturer revenues and operating costs, many other elements must be considered. Indeed, the better the concept is in terms of passengers' comfort, the higher the ticket price can be compared to competitors. The better the vehicle is designed, the lower the operating

costs will be. Nevertheless, there is usually a trade-off between acquisition and operating costs. This trade-off is often translated into a conflict of interest between manufacturers and operators. While the revenues are mainly driven by market trends, this work will focus on cost rather than revenue optimization. Figure 40 presents a description of the different categories of life-cycle costs and their main components.

RDT&E costs includes technology research, design, software, and system engineering as well as the costs related to the prototype fabrication, flight, evaluation, and testing. RDT&E costs also include all certification-related costs. These costs are fixed and do not depend on the number of vehicles produced.

Production costs correspond to the costs required to manufacture and assemble every vehicle subsystem such as airframe, engines, avionics, and thermal protection system. This covers labor, production tooling, machines, materials as well as the overhead and administrative expenses. Contrary to RDT&E costs, production costs are variable and the number of vehicles produced will greatly impact them.

Total operating costs are usually divided into three main categories: Direct Operating Cost (DOC), Indirect Operating Cost (IOC), and Refurbishment/Spares Cost (RSC). DOCs are defined as costs related to general flying operations. They include flying operations, fuel, propellants, insurance, maintenance, maintenance burden, depreciation, interest, and rents. IOCs are defined as costs required to operate the vehicle but not related to the flights themselves. They include landing and navigational fees, station costs, passenger-related costs, advertisement and promotion expenses, and administrative costs. RSCs can be defined as the costs required to change the critical vehicle components in addition to typical maintenance operations. For aerospace vehicles, typical components are the thermal protection system, key components within the rocket engine, etc.

Disposal costs are the costs incurred to dispose of the vehicle after its useful life in accordance with regulatory and legal environments. They include transportation, disassembly, and destruction costs. Even though they are often negligible compared to the other costs and highly depend on the second life of the vehicle, they are translated into requirements. For example, some toxic materials cannot be used. Thus, they will not be taken into account

in this study.

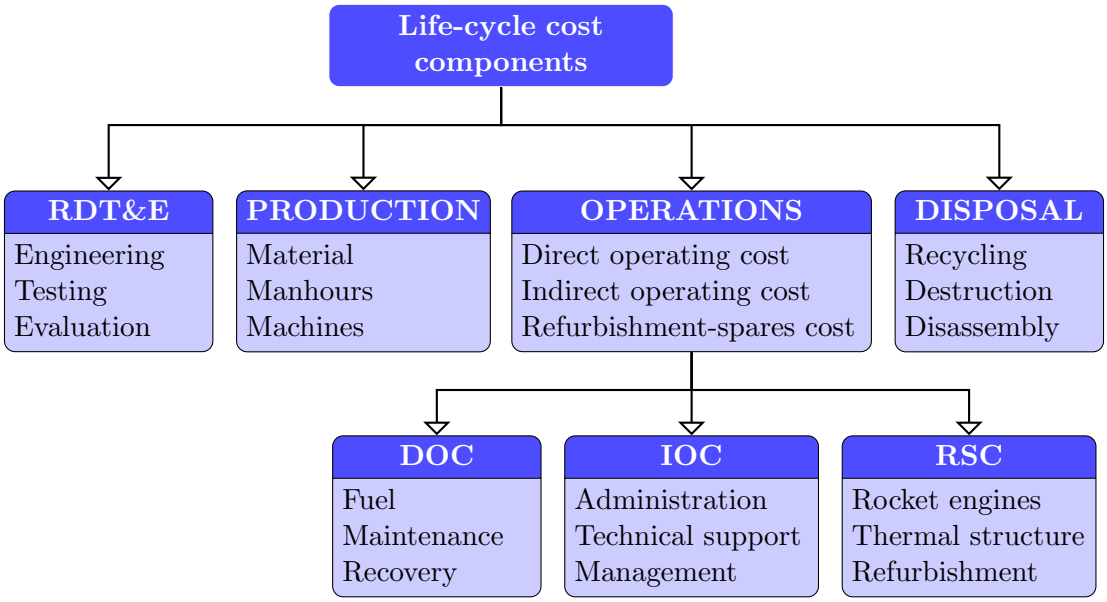



**Figure 40:** Life-cycle cost components

#### *3.3.6.2 Overview of Cost Estimation Techniques*

The broadness of the cost components led to the emergence of a large number of approaches. However, a categorization of these approaches is prone to controversy and inconsistency. T. Farineau et al. [141] classify the techniques into three different categories: analytic, analogic, and parametric methods. R. Roy [371] divides these techniques into five categories: traditional cost estimating, parametric cost estimating, feature based costing, neural network based cost estimation, and case based reasoning. A. Niazi et al. [318] provide a more detailed decomposition of the Product Cost Estimation (PCE) techniques as illustrated in Figure 41.

Finally, O. Trivailo et al. [443] describe the different categories of existing tools within the aerospace domain with a focus on early design phase applications. Their categorization includes six approaches: expert judgment, parametric modeling, rough order of magnitude, loose analogy, close analogy, and engineering build-up. Based on the previous studies, a new categorization, detailed enough for the purpose of this study, is presented in Figure 42 and will be followed for the remainder of the description. Four main categories are identified:

expert judgment, analogy, parametric modeling, and analytic modeling.

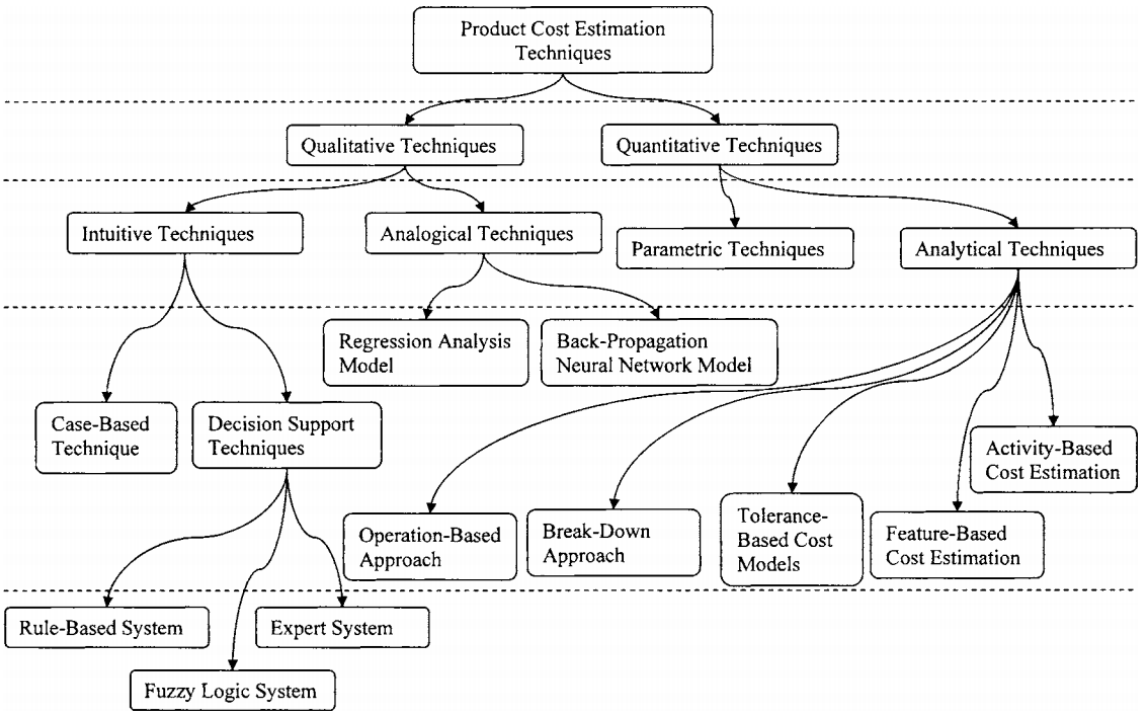



**Figure 41:** Classification of the PCE techniques [318]

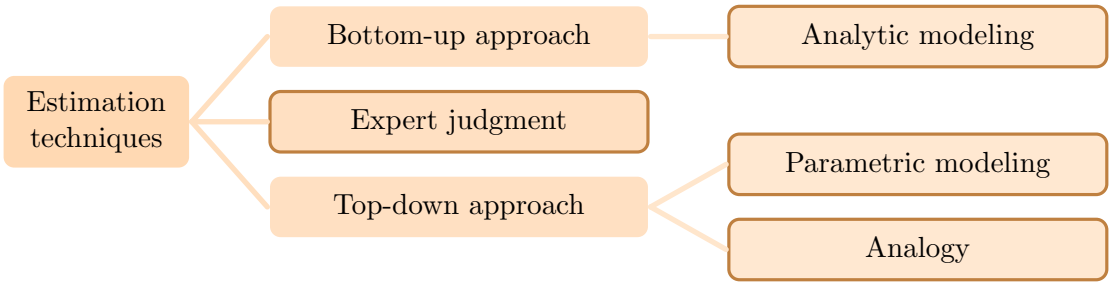



**Figure 42:** Overview of the different estimation techniques

The expert judgment method is based on a group of experts who use their own knowledge and experience to predict the cost. They are usually supported by a large database in which they can select the data relevant to their specific problem. This method is often considered very subjective because it depends on the specific people in charge of the task. Moreover, the

results could be biased by either political or business pressures. In spite of these drawbacks, this method remains widely used in today's industry. Besides, thanks to its flexibility, it can be used throughout the entire design process and no precise and pre-defined variables are required. An example of the implementation of such methods is the Delphi method. The latter collects the opinion of experts through questionnaires in order to statistically reach an expert consensus [198].

The analogy method extrapolates the cost by comparison with similar existing concepts. This method is only based on a single or few data points and consequently correction factors must be applied to correct for differences in size, technology level, complexity, etc. Various elements of comparison can be used such as stakeholders' specifications, performance, vehicle size, similarities in the architecture, etc. O. Trivailo et al. [443] identify two subcategories: loose analogy and close analogy. While the first one only requires few "loosely similar" characteristics, the second one requires more "closely similar" characteristics. Contrary to close analogy, loose analogy is usually adjusted with scaling factors and is used in early design phases. In general, this method is very fast, cheap to implement, and transparent for users. Nevertheless, it still relies on subjective considerations to determine both concepts similar to the one studied and scaling factors. Moreover, detailed economic data must be available for the similar concept(s). An example of such method is the Case-Based Reasoning (CBR), which has been implemented into Kate from Acknosoft and ReCall from iSoft [175, 471].

The parametric method consists in a series of mathematical relations called Cost Estimating Relationships (CERs) that estimate the cost with respect to its major parameters. Frequent cost drivers are weight, maximum Mach number, material, etc. They are established using historical data and statistical models and can be calibrated for new technologies or other factors. Such models exist under various forms (linear, power, exponential, polynomial, etc.) with one or more variables. This technique is one of the most widely used in the aerospace industry, especially in early design phases. Its usefulness is also confirmed by the fact that the FAA accepts these CERs for proposal preparations [443]. Moreover, many

famous software use these relationships such as the Statistical-analytical Model for Cost Estimation and Economic Optimization of Space Transportation Systems (TransCost) [239], NASA Air Force Cost Model (NAFCOM) [286], and Development And Procurement Cost of Aircraft (DAPCA) from RAND Corporation [201]. Nevertheless, even if this top-down approach is fast, cheap, easy to use, and only requires few input variables, it still has some drawbacks. Indeed, two main difficulties arise: one concerning the establishment of the CERs and another concerning their validity. Indeed, in order for good models with good correlations to be created, the quality of the database on which they rely must be good enough and must be representative of the problem studied. Besides, even if they have been successfully established, frequent changes in manufacturing processes, materials, technologies, and economic environment make the use of these CERs difficult, or even impossible for new and innovative architectures. An example of CER is provided in Equation 53 to estimate the development cost $\$_D$ of commercial aircraft [354], where $W_e$ is the empty weight and $V$ the maximum velocity.

$$\$_D = 48.5 W_e^{0.63} V^{1.3} \tag{53}$$

The analytic modeling, or engineering build-up method decomposes the project into tasks and components. Once fully decomposed, the cost of each subsystem can be assessed by the corresponding experts using the specific characteristics of each subsystem and customized cost estimation methods. This approach is usually used during the late design phases when a detailed description of the concept is known and clearly defined. Indeed, while being very accurate, this technique requires a lot of resources and precise data about the product, its manufacturing process, materials, etc. This method is usually applied to fix the product price at the end of the design and manufacturing processes.

Table 18 compares the efficiency of each cost estimation method with respect to each design phase. Even if expert judgment can always be applied, it is not suitable for this project. Analytic modeling requires too much information about the project and the concept to be applied at conceptual design level. For automation purposes and avoidance of man-in-loop requirements, parametric modeling is preferred. Existing tools leveraging this

approach are discussed in the following section.

**Table 18:** Applicability of Cost Estimation Methods throughout the design process

| | Conceptual | Preliminary | Detailed |
|---|---|---|---|
| **Expert judgment** | ✓ | ✓ | ✓ |
| **Analytic modeling** | | | ✓✓ |
| **Parametric modeling** | ✓✓ | ✓✓ | ✓ |
| **Analogy** | ✓ | ✓✓ | ✓ |

#### *3.3.6.3 Overview of Cost Estimation Tools and Models*

Parametric modeling techniques have been implemented into a large number of tools and can be compared against various criteria: level of detail (number of inputs/outputs), phases of the life-cycle costs that are covered, availability, easiness to use, and configurations covered (launch vehicles, aircraft, etc.).

Space Transportation Systems Cost Estimation and Economic Optimization (TransCost) is a publicly available tool developed by D. E. Koelle in 1971 and frequently updated. The last version of the CERs has been is the TransCost Version 8.2 from 2013 and are documented in [240]. This tool only concerns launch vehicles (re-usable and expendable) and covers all life-cycle costs. The models use Man-Year as cost unit which is totally independent of inflation and currency exchange rates. The CERs are mainly driven by mass variables, subsystem type, and complexity factors. Moreover, it is well documented and the equations are transparent for the users so that it can be implemented in all integrated environments [239].

Unmanned Space Vehicle Cost Model (USCM) was developed by the United States Air Force Space Division. It only concerns unmanned, earth-orbiting space vehicles. It is able to predict flight hardware, RDT&E, production, and operating costs. It provides models for subsystems at a component level. Nevertheless, this tool became an ITAR tool and is consequently not available [440].

Advanced Missions Cost Model (AMCM) was developed by the Exploration Programs

Office of NASA's JSC. Unlike most of the other static CERs, this model determines a pattern of cost across the entire space composed of the various cost parameters. Thanks to this multidimensional perspective, extrapolations outside the calibration population of data point is more accurate. As such, this model uses complexity factors instead of linear relationships. It is able to estimate the cost of various types of concepts: planetary spacecraft, manned re-entry, missiles, commercial aircraft, fighter aircraft, etc. Nevertheless, its flexibility requires the users to calibrate the model and is therefore complicated to handle. This tool also became ITAR and is not available [389].

Advanced Cost Estimating System (ACES) was developed by the German 4cost company. Contrary to the other tools, it does not have a database of past mission concepts. Instead, it uses multidisciplinary database to create the CERs. It has been used by the DLR, EADS Astrium, OHB, and MT Aerospace. The inputs it takes are relatively different when compared to the other tools and are more related to economic conditions, manufacturing processes, and development strategies. In addition to be relatively expensive, this tool cannot support architecture comparison and configuration optimization since it does not input these types of design variables [2, 443].

Launch Vehicle Cost Model (LVCM) was developed by the Department of Defense in a classified tool governed by strict ITAR regulations. Its purpose is to estimate RDT&E as well as operation and support costs for launch vehicle programs. Inputs of this tool are very detailed and need to be entered by a well-trained user [443].

The NASA/Air Force Cost Model (NAFCOM) was developed by the Science Applications International Corporation (SAIC). It is an Excel-based tool for space hardware that can be applied at subsystem and component levels. The tool uses weight, materials, and complexity factors as primary cost drivers and is able to predict RDT&E, production, and operating costs. It has been established using a database of more than 120 reference projects and 8 of these are manned spacecraft. The software allows the users to select among two estimation options: typical parametric CERs or specific analogy. Nevertheless, this tool is ITAR and cannot be used [286, 308].

The Development And Procurement of Costs of Aircraft IV (DAPCA IV) consists in

a series of freely available CERs developed by the RAND Corporation [201]. It estimates the number of hours required for engineering, tooling, manufacturing, and quality control. Once estimated, the use of appropriate labor rates allows users to compute the total cost. It also provides equations for manufacturing material cost. In addition to this airframe-dedicated tool, engine costs can be estimated using other RAND Corporation's studies [39] and studies from Hamburg-Harburg University [390].

The ALCCA module integrated into the FLOPS is a code developed by NASA and the Georgia Institute of Technology. It estimates aircraft manufacturing, production, RDT&E, and operating costs for fighter, commercial, and GA aircraft. It also allows users to plot the manufacturer and airline cash-flows. It uses weights and performance calculations from FLOPS and specific factors about the economic context and the complexity of the aircraft. This tool is available to the author.

The PRICE cost model, developed by PRICE Systems, is a series of modules with different purposes: PRICE-H, which estimates hardware development and production costs, PRICE-HL, which predicts hardware maintenance and support costs, PRICE-S, which assesses all types of software-related costs, PRICE-M, which estimates costs of electronic components, and PRICE-PM, which schedules projects. Hence, this model is not dedicated to space systems and requires a lot of detailed key inputs in order to produce accurate results. Moreover, it requires a relatively long user training due to the complexity of the tool. It is widely used in the aerospace industry but its scope is too broad to be of interest at this stage of the design process [347].

Galorath Inc. developed the Systems Evaluation and Estimation of Resources (SEER), which is a series of tools similar to PRICE. Hence, SEER-H is devoted to hardware, electronics, and systems, SEER-MFG is designed for fabrication and assembly and other platforms are dedicated to IT and software. The estimations are mainly based on sector-specific models and require a significant amount of work from the users to be calibrated for a specific scope. Estimates can be obtained at different levels: from component to system of system level [166].

MAP-H was developed by the French company 3f to assess the hardware cost of a large

variety of products in many fields: aerospace, chemistry, industrial, etc. It is able to compute all life-cycle costs as well as the possible integration costs for large systems of systems. This universal cost estimation tool is based on a series of specific powerful algorithms that require to be calibrated and adjusted by the users. The variables required by the algorithms are also dependent on the configuration selected by the users but they must describe the product, its level of complexity, its context, and its operating environment. Due to its high flexibility, this commercial tool also requires a long training and is not easy to use [1].

R. A. Goehlich developed a statistical-analytical model called Suborb-TransCost, whose applications only include suborbital vehicles [176]. The model, adapted from the classical TransCost model, is organized in four interconnected submodels: development cost, vehicle cost, total operating cost, and total profit. It is perfectly adapted to suborbital vehicles and covers all configurations considered in this study: winged and ballistic vehicles, with optional engines, etc. Nevertheless, it only addresses the cost of liquid engines.

**Table 19:** Comparison of the various cost estimation tools

| | **All rocket engines** | **Available** | **Easy** | **All concepts** | **Appropriate level of detail** |
|---|---|---|---|---|---|
| **TransCost** | ✓ | ✓ | ✓✓ | ✓ | ✓ |
| **USCM** | | | ✓ | | ✓✓ |
| **AMCM** | ✓ | | | ✓✓ | ✓✓ |
| **ACES** | | ✓ | | ✓ | |
| **LVCM** | | | | ✓✓ | |
| **NAFCOM** | ✓ | | ✓✓ | ✓ | ✓✓ |
| **DAPCA IV** | | ✓✓ | ✓✓ | | ✓✓ |
| **ALCCA** | | ✓✓ | ✓ | | ✓ |
| **PRICE** | | | | ✓✓ | |
| **SEER** | | ✓✓ | | ✓✓ | |
| **MAP-H** | | | | ✓✓ | |
| **Suborb-TransCost** | | ✓✓ | ✓✓ | ✓✓ | ✓ |

As shown in Table 19, **no existing tool currently meets all requirements. Suborb-TransCost seems to be the most suitable for our application. Nevertheless, this model only discusses liquid rocket engines and the level of detail is not high enough to enable any optimization.**

Since the rocket engine is a key cost driver, accurate trend identification requires a more precise modeling of the propulsion system. In addition to TransCost, for solid and liquid engines and Suborb-TransCost for liquid engines, only one other available methodology has been found. Developed by Sjovold and Morrison, CERs are provided for liquid and solid rocket engines as a function of their main design parameters [185]. **However, no existing model has been found to assess the cost of hybrid rocket engines.** These observations result in the formulation of the following Research Question:

**RESEARCH QUESTION 3.2: How can life-cycle costs of vehicles powered by hybrid rocket engines be evaluated in early design phases?**

To address this question, existing models for solid and liquid engines will be modified to enable the modeling of hybrid engines. The lack of historical data motivates the development of a more physics-based approach that can be successfully implemented to model all life-cycle costs of hybrid rocket engines. The nature of the hybrid engine comes from the fact that it is composed of elements from both liquid and solid engines. As shown in Figure 43, **a hybrid engine can be perfectly decomposed into two parts: one solid engine and one liquid engine without nozzle. Hence, existing CERs for liquid and solid engines can be used to model life-cycle costs of hybrid engines.** CERs from Sjovold and Morrison model production costs as a function of key engine variables and development costs as a function of production cost.

Thus, the production cost of a hybrid rocket engine can be modeled as the sum of these two components based on the following four-step methodology once the engine has been sized:

1. Extract the cost of the nozzle from the overall liquid engine cost using historical data.

2. Compute the production cost of the liquid sub-engine without its nozzle using CERs from Sjovold and Morrison.

3. Compute the production cost of the solid sub-engine using CERs from Sjovold and Morrison.

4. Sum the costs of these two components.

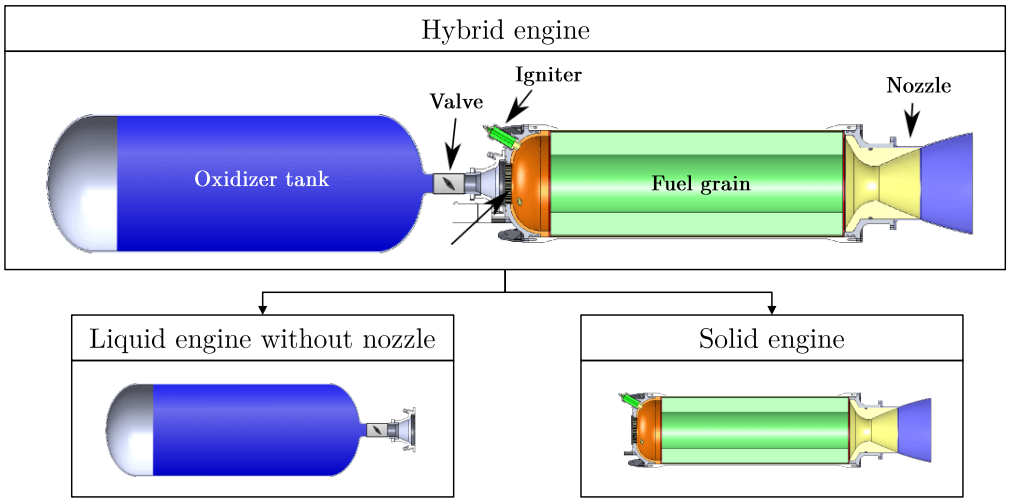



**Figure 43:** Physical decomposition of hybrid engines

To predict the RDT&E cost of hybrid rocket engines, RDT&E costs for the solid and the liquid engines with the corresponding production cost will be computed. Then, RDT&E cost for the hybrid engine is assumed to be the weighted average of these two cost based on the relative weight of each sub-engine.

Finally, operating costs will be decomposed into two categories: maintenance and propellant. While maintenance costs follow the same principle as RDT&E costs, propellant cost will be directly computed using the engine performance and the propellant cost. Therefore, wisely combining CERs for liquid and solid rocket engines can provide good estimates for hybrid engine life-cycle costs. These observations lead to the formulation of the following Hypothesis:

**HYPOTHESIS 3.2: IF a hybrid rocket engine is physically decomposed into a liquid engine without its nozzle and a conventional solid engine THEN its life-cycle costs can be predicted using existing models developed for liquid and solid engines.**

In order to validate this Hypothesis, Experiment 3.2 is implemented. As inputs, it requires a description of existing hybrid rocket engines. The experiment consists in computing the life-cycle cost of these engines using the new module and compute the life-cycle costs for solid and liquid engines with the same requirements. As no actual data are available, the validation of Hypothesis 3.2 is based on the comparison with the cost of other types of engines. In particular, one must check if, for a given set of requirements, the life-cycle costs of hybrid engines are between the ones of solid and liquid engines.

While life-cycle costs represent one of the most important decision criteria, safety is also crucial, especially when dealing with commercial flights. Hence, the next section discusses risk analysis.

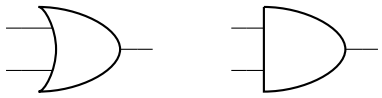
### 3.3.7 Safety Analysis

Safety analysis takes an important part in current decisions when designing innovative aerospace vehicles, especially for vehicles involving commercial passengers. Thus, to evaluate the safety level of each possible design, it is necessary to develop an accurate risk and reliability assessment module. The goal is to evaluate the safety level of each possible design. In this section, four different reliability and risk assessment techniques currently applied in aerospace design are investigated [464]: Fault Tree Analysis (FTA), Reliability Block Diagrams (RBD), Markov analysis, and Petri Net analysis. These techniques are then compared, evaluated, and leveraged to find the best strategy for this research. They can be decomposed into two different types of analysis. While the first two methods perform a static analysis, the last two perform a state space one.

#### *3.3.7.1 Fault Tree Analysis*

An FTA is a top-down or deductive approach to reliability modeling. This method identifies the top-level failure (failure dramatic for the vehicle) along with the potential causes [251, 372, 412]. The lower levels are composed of events that cause the upper levels. Those levels are linked to each other using logic gate links that represent the causal effects. The top of the tree is composed of the main events, while the lowest level is called basis level and is defined by all the basis events. The progression from a basis event to the main event is called a scenario. The higher the number of scenarios defined, the more real the model is. All the events have to be independent from each other. Typical FTA versions only have two logic gates: AND and OR, as described below:

- Gate OR: The top event occurs if at least one of the down events occurs. The gate OR is represented by the symbol displayed in Figure 44(a).
- Gate AND: The top event occurs if all the down events occur. The gate AND is represented by the symbol displayed in Figure 44(b).



(a) Gate OR (b) Gate AND

**Figure 44:** Illustration of OR and AND gates

If no intermediate or basis events are repeated, reliability calculations can be done directly using boolean operations. Inputs can be either total probability, or failure/success rates. Equations 54 and 55 show how to compute the probability of the event for a gate AND and OR, respectively, where $P_i$ are the probabilities of the different inputs.

$$P_{AND} = \prod_i P_i \tag{54}$$

$$P_{OR} = 1 - \prod_i (1 - P_i) \tag{55}$$

However, intermediate or basis events can take part into different scenarios. Hence, in order to keep the calculation straightforward, repeated events have to be avoided. Therefore, the FTA needs to be restructured, leading to a more compact representation. However, while the impact assessment of each event is more convenient, the failure scenarios are less explicit. Figure 45 displays an example of a tree transformation with a repeated event A.

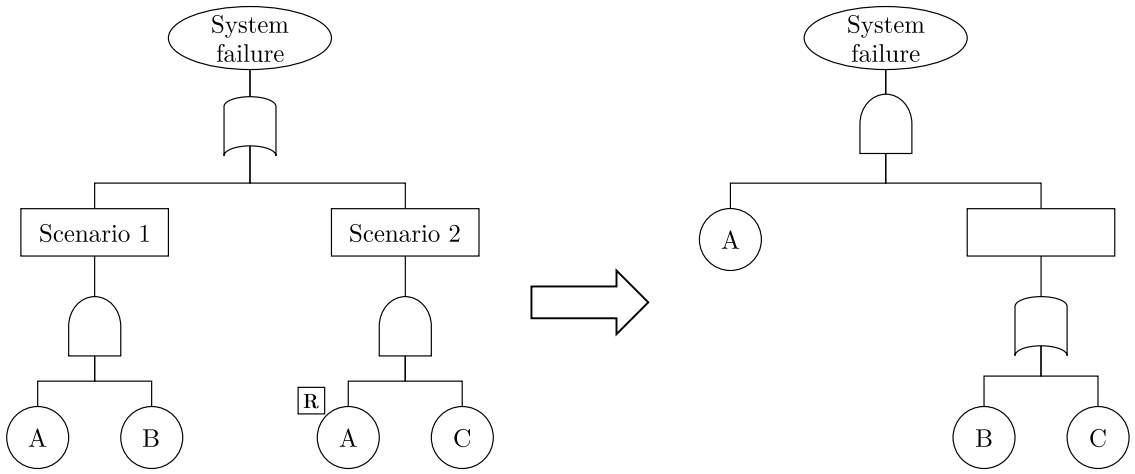



**Figure 45:** Transformation of an FTA to avoid repeated events [464]

FTA can become rather complicated when applied to complex systems. In particular, this technique was used for the Space Shuttle and its FTA has more than 1,400 elements [488]. Nevertheless, FTA is commonly used by companies since it has a relatively intuitive and simple representation and can perform easy calculations using Boolean operators.

#### *3.3.7.2 Reliability Block Diagrams*

RBD is a static tool very similar to the FTA [116, 140, 297]. The subsystem failure interactions are modeled and each subsystem is represented in a single block with an associated failure rate. Inputs are on the left while outputs are on the right. Each block represents a subsystem to which a failure rate is assigned. The blocks are connected either in series or in parallel. A parallel configuration represents a subsystem redundancy in the risk analysis. Figure 46 represents two parallel systems connected in series. For this configuration, the

set of possible paths is: $\{A, C\}$, $\{A, D\}$, $\{B, C\}$, and $\{B, D\}$.

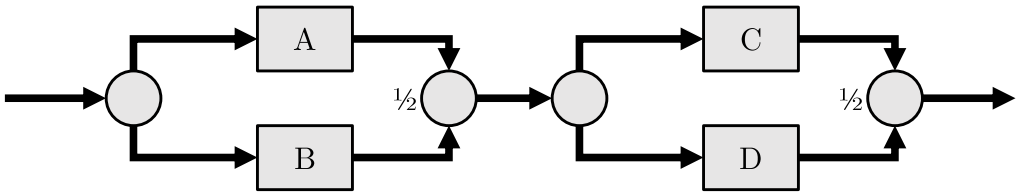



**Figure 46:** Reliability Block Diagram [464]

*3.3.7.3 Markov Analysis*

A Markov analysis is a method used in engineering to evaluate system reliability [127, 162, 350]. While FTA and RBD are static models imposing the independence of events, the Markov analysis does not need independence and is a truly dynamic state space model. A Markov analysis consists in a series of Markov chains, made of states and transitions between the states. A state is the condition in which the element is currently operating, and is set as failed or operated for a reliability analysis. Transitions between states are usually described as failures and repairs of the system. Figure 47 displays an example of a simple Markov chain where the transitions occur with a failure rate $\lambda$ and a repair rate $\mu$.

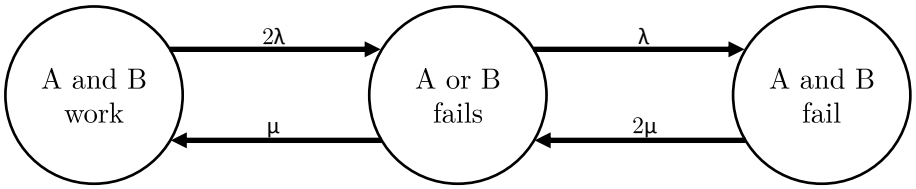



**Figure 47:** A parallel system with two identical units [488]

The Markov chain requires the system to have the memoryless property: the future states of the problem are independent of the past states. The probability of making a transition must be constant with time, which means that the likelihood of failure is only dependent on the working state of the component. One major drawback of the Markov analysis is that the model size increases exponentially with the number of components.

The diagram for a Markov analysis becomes very complicated even with a relatively simple system. One solution to this problem is to use a Markov analysis for parts of the system that are interdependent and then use the resulting subsystem failure probability as

an input into a model, which can handle larger systems such as FTAs and RBDs.

*3.3.7.4 Petri Net Analysis*

A Petri net is a reliability method first proposed by C. A. Petri in 1962 [7, 223, 304, 336, 337]. It provides the means to analyze the dynamic behavior of systems. A Petri net has two types of nodes: a place node and a transition node. In a Petri diagram, arcs connect places and transition nodes. One or more tokens can exist in each place determining the state of the system. A transition occurs when every input to that transition has at least one token in it. The "firing" occurs by removing one token from the input and placing that token into the output. An illustration of the Petri diagram of a repairable unit is provided in Figure 48.

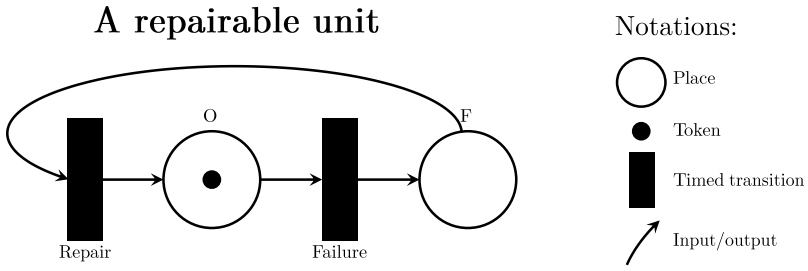



**Figure 48:** Illustration of a repairable unit [488]

Petri net can be used to replicate the work of an FTA. It is much more powerful in determining reliability as they can model dynamic systems.

*3.3.7.5 Method Selection*

Based on the previous analysis, Table 20 summarizes the key advantages and drawbacks of each method. As described in Table 20, top-level oriented techniques are more suitable for a conceptual design level. Indeed, Markov and Petri Net analyses require detailed data about the vehicle and are very complex. In addition, the FTA offers a better visualization tool than the RBD. Consequently, safety analysis will be conducted using a dynamic and upgradable FTA that accounts for all possible configurations.

**Table 20:** Safety and reliability assessment methods

| | **Top-level oriented** | | **State-space oriented** | |
|---|---|---|---|---|
| | **FTA** | **RBD** | **Markov analysis** | **Petri Net analysis** |
| **Advantages** | Classical boolean calculations. Simple and good visualization capabilities. | | Truly dynamic modeling for dependent events. | A token can have associated continuous counters, which enables the remembering of its age. |
| **Drawbacks** | Scenario-based (all relevant scenarios must be explicitly described). Static configuration: poorly model dynamic and adaptive systems. | | Exponential increase of the size of the model with respect to the number of modeled units. Time-independent failure rates (no aging). | Very complex and rarely used in reliability assessment. |

Sections 3.3.2 to 3.3.7 addressed both flying, economic, and safety performance of the vehicle. Once fully integrated into the design framework, the different modules will enable designers to evaluate the performance of each alternative. The establishment of this design framework provides the inputs required to implement Experiments 1 and 2. This implementation, which helps improve the decision-making process, is detailed in the next section.

## *3.4 Step 4: Make Informed Decisions*

Decision-making is defined as "the act of reaching a conclusion or of passing of judgment on an issue under consideration" [429]. Two main decision-making approaches can be distinguished: heuristic and analytic. The first approach is based on experience and personal judgment while the second one is built on quantitative information, models, and mathematical algorithms. The analytic approach will be preferred since it provides a sense of optimality instead of satisfaction only. This results in a need for rigorous and systematic processes, as developed in Chapter 2. In particular, for emerging market, decision-making

attempts to:

- Optimize the vehicles against multiple objectives.
- Design vehicles under evolving uncertainty in requirements.
- Allow decision makers to freeze requirements and prioritize their objectives.
- Perform trade-offs between constraints and objectives.

Based on the two processes developed in Sections 2.1 and 2.2, the development of the decision-making framework follows the following steps:

1. Architecture comparison and optimization
   (a) Develop the optimization algorithms (NSGA-II) for each architecture
   (b) Create the architecture-based evolutionary algorithm
2. Uncertainty modeling
   (a) Model uncertainty with time-dependent membership functions
   (b) Propagate membership functions using fuzzy logic theory
3. Objective prioritization
   (a) Develop a DoE using mean values and degrees of uncertainty of each requirement
   (b) Develop a TOPSIS based on Pareto frontiers given by the previous optimization process

The implementation of this framework will provide the required capabilities to execute three of the experiments developed in Chapter 2. First of all, the capability of the new optimization algorithm will be tested through Experiment 1.2. By combining this experiment with Experiment 1.1, Experiment 1 can also be easily implemented. It will ensure that the sequential combination of the alternatives generation methodology and the optimization process really improves design space exploration. Finally, Experiment 2 can be set up to validate the ability of the methodology to address evolving uncertainty in requirements. The overall research structure is displayed in Figure 49.

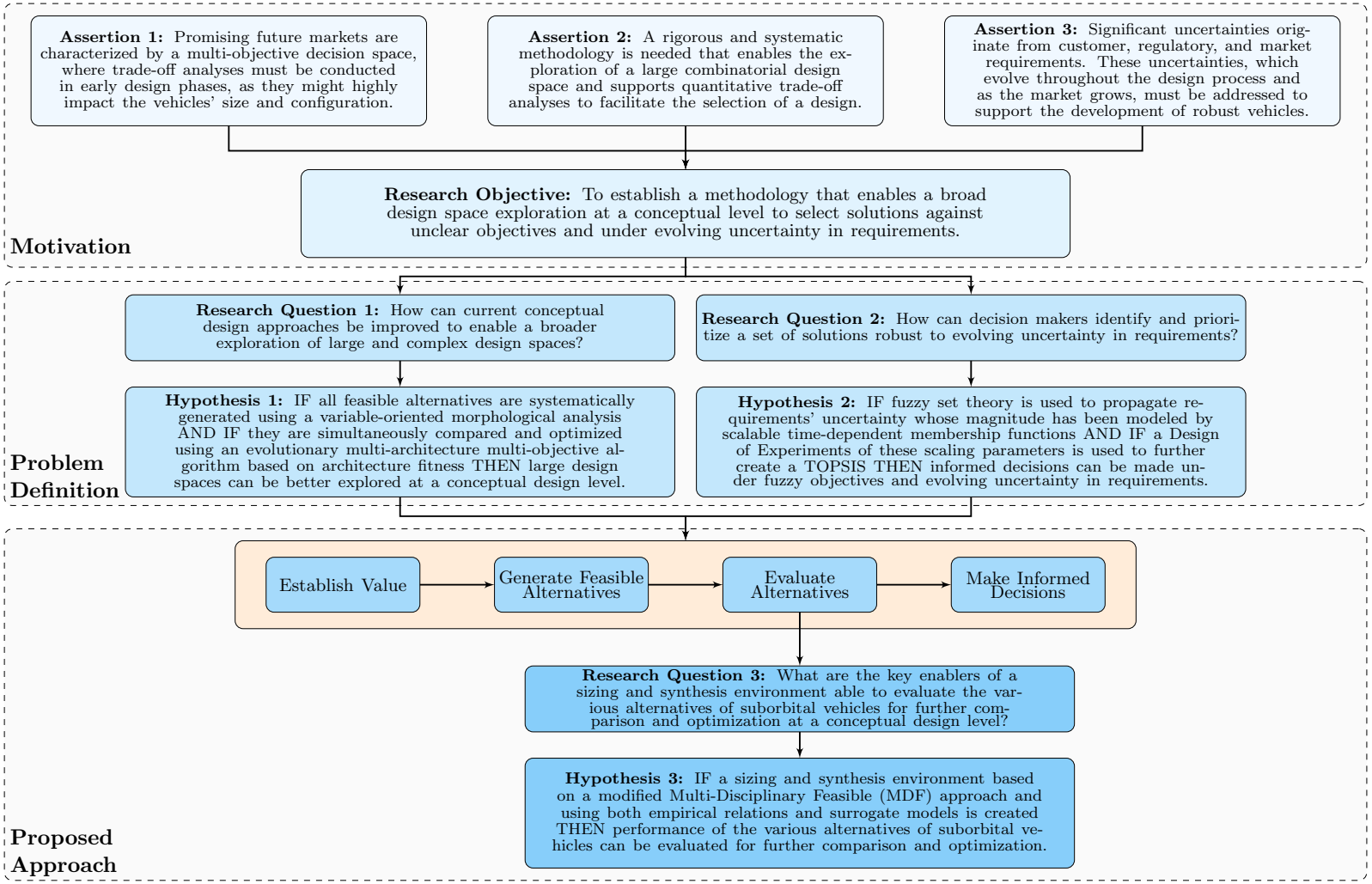



**Figure 49:** Overall structure of the research

As mentioned in Section 3.1, the proposed decision-making process follows a rigorous, structured, and comprehensible four-step process that will be detailed in the next chapters. It starts with the identification of the decision criteria, as discussed in the following chapter.

# CHAPTER IV

# STEP 1: ESTABLISH THE DECISION CRITERIA

This chapter discusses the establishment of the various criteria that will both limit the available design space and be used to evaluate and compare the various design alternatives. Indeed, at the beginning of each project, there is a need for mapping the customer requirements to quantifiable and precise criteria. First, the design objectives that can be used to evaluate alternatives are clearly defined. Then, the design constraints that limit the design space are presented, along with their uncertainty model.

## *4.1 Design Objectives*

In order for space tourism to really materialize, designers need to focus on the actual desires and motivations of potential customers, while also addressing safety and certification issues. The Socotec Group surveyed potential customers about their interests and obstacles in suborbital flights [252]. The results are presented as a ranking in Table 21.

**Table 21:** Ranking of interests and obstacles in suborbital flights [252]

| | **Interests** | **Obstacles** |
|---|---|---|
| **1** | See the Earth from space | Too expensive |
| **2** | Experience rocket acceleration | Interest in other activities |
| **3** | Get a feeling of what astronauts experience | No interest in space |
| **4** | Experience micro-gravity | Too risky |

Based on these results, two main obstacles appear to be addressable by designers and must be used as design objectives: life-cycle costs and safety. Indeed, in order to overcome some of the obstacles, a safe and affordable vehicle must be designed. In addition, to ensure the continued existence of the suborbital market, there is a need for satisfying paying passengers. For that purpose, the last criterion will be passenger experience. Each of these three criteria are detailed in the following sections.

### 4.1.1 Affordability

In current economic conditions, designing an economically viable product is crucial. Not only must the vehicle be affordable to develop and to build, but it must also be cheap to operate. To quantify all these aspects, the total program cost is chosen as the metric to quantify affordability and is minimized. Equation 56 relates the vehicle's affordability $y_a$ to the different economic metrics defined in the design framework. In this equation, $C_D$ is the development cost of the vehicle, $n_v$ the number of vehicles built, $C_M$ the manufacturing cost of the vehicle, $n_l$ the number of launches per vehicle over the entire program, and $C_O$ the total operating cost of a single vehicle for a single mission.

$$y_a = C_D + n_v \left( C_M + n_l C_O \right) \tag{56}$$

### 4.1.2 Net Present Value

The Net Present Value (NPV) is the key metric commonly used by managers and high executives to perform project valuation and make investment decisions. It is representative of the economic viability of a project and numerous investors use it to evaluate their projects of interest [165]. The NPV is the sum of the discounted Free Cash Flows (FCFs) of the project. Discounting enables the concept of time value of money, which accounts for the fact that the later the money is received or spent, the less it is worth. Indeed, money received early can be invested and increase in value later on. Discounting also accounts for the riskiness of the market. Equation 57 provides the definition of the NPV of a project lasting $N$ periods, having cash flows $FCF_k$ at each period $k$, and with a discount rate $i$.

$$NPV = \sum_{k=1}^{N} \frac{FCF_k}{(1+i)^k} \tag{57}$$

The NPV presents many advantages. As previously stated, it considers the time value of money. Moreover, it provides managers with a metric that quantifies the profitability of a project. As a consequence, it can be used to compare and rank different projects with respect to their economic performance. Indeed, the sign of the NPV indicates whether a project is profitable enough or not. Indeed, a positive NPV indicates that the projected

earnings (in present dollars) generated by a project or investment exceed the anticipated costs (in present dollars). Similarly, a negative NPV results in a net loss. Hence, the Net Present Value Rule states that an investment should only be made if the NPV is greater that zero [353].

### 4.1.3 Safety

Since suborbital vehicles are designed to carry people, the most important criterion is their safety. Indeed, similar to aircraft or other transportation systems, the failure rate has to be smaller than a fixed value in order for the vehicle to be certified. As there is no existing standard and the calculation requires extremely precise information, safety cannot be evaluated using failure rates. Instead, an arbitrary quantitative scale $y_r$ representing the risk level is defined, as presented in Section 6.7. Therefore, the objective is to minimize this risk level $y_r$.

### 4.1.4 Passenger Experience

Since no real data is available for space commercial passengers, the requirements are extracted from commercial flights. Many studies have been conducted that rank passenger personal space as the top priority for passengers [78, 156, 301, 486]. In particular, the APEX showed that the major preoccupation for passengers is legroom [16]. As a consequence, increasing the seat pitch can be used as a representative objective to improve passenger experience.

In addition, the time spent in microgravity is one of the unique aspects of a suborbital flight, especially for the market established around the scientific research. Hence, the passenger experience can also be modeled by the duration of the weightlessness phase. The latter starts once the rocket engine is shut down and ends at the beginning of the reentry. The reentry altitude is around 30 km for a slender body and 45 km for a winged body. In order to determine the duration of the weightlessness phase $t_w$, it is assumed that the only external force that acts on the vehicle is the Earth gravity $g_0$. This assumption is valid at such high altitudes since the atmospheric density is negligible. As a consequence, applying the Newton's Second Law on the vehicle results in Equation 58, where $\vec{a}$ is the vehicle's

acceleration.

$$\vec{g_0} = -\vec{a} \tag{58}$$

Using only the vertical component, Equation 59 can be derived, where $h_{w0}$ is the initial altitude of the weightlessness phase, $v_{w0}$ the vertical velocity of the vehicle at the beginning of the weightlessness phase, and $h_{wf}$ the altitude at the beginning of the reentry.

$$-\frac{1}{2}g_0 t_w^2 + v_{w0} t_w + h_{w0} - h_{wf} = 0 \tag{59}$$

The vertical speed at the beginning of the weightlessness phase can be calculated as a function of the maximum altitude $h_{max}$ using Equation 60, where $t_{hmax}$ is the time required to reach the maximum altitude.

$$\begin{cases} -\frac{1}{2}g_0 t_{hmax}^2 + v_{w0} t_{hmax} + h_{w0} - h_{hmax} = 0 \\ v_{w0} = g_0 t_{hmax} \end{cases} \tag{60}$$

The resolution of the previous system of equations results in Equation 61, which provides the vertical velocity of the vehicle at the beginning of the weightlessness phase.

$$v_{w0} = \sqrt{2g_0 \left(h_{max} - h_{w0}\right)} \tag{61}$$

Combining Equations 59 and 61, the total duration of the weightlessness phase can be calculated, as shown in Equation 62.

$$t_w = \sqrt{\frac{2}{g_0}} \left( \sqrt{(h_{max} - h_{w0})} + \sqrt{(h_{max} - h_{wf})} \right) \tag{62}$$

While the optimization aims at improving these design objectives, there is a need to limit the design space based on regulations and customer requirements, as discussed in the next section.

### *4.2 Design Constraints*

In order to ensure the feasibility and the viability of the optimized design, several constraints must be taken into account in various domains. Using existing analyses [252, 377, 418], three main constraints have been identified: the maximum altitude reached by the

vehicle $h_{max}$, the maximum load factor $n_{max}$, and the total number of passengers per year. The following sections discuss the initial membership functions used to model each constraint and describe time-dependent models that can be applied to account for the evolving nature of uncertainty.

### 4.2.1 Initial Membership Functions

#### *4.2.1.1 Maximum Altitude*

In order for passengers to be considered as space tourists, the maximum altitude has to be, at least, 100 km. However, some of the customers might want to reach a higher altitude, and consequently have a longer time in microgravity. Some applications such as the launch of small satellites or scientific research might require higher/lower altitude. Therefore, there is a large amount of uncertainty on this constraint. For example, the Lynx developed by XCOR only reaches an altitude of 60 km [344], while the Crusader X designed by Micro-Space, Inc. reaches an altitude of 120 km. However, most of the current concepts aim at reaching an altitude between 100 and 110 km. Hence, the uncertainty is modeled by a trapezoidal membership function, as presented in Figure 50.

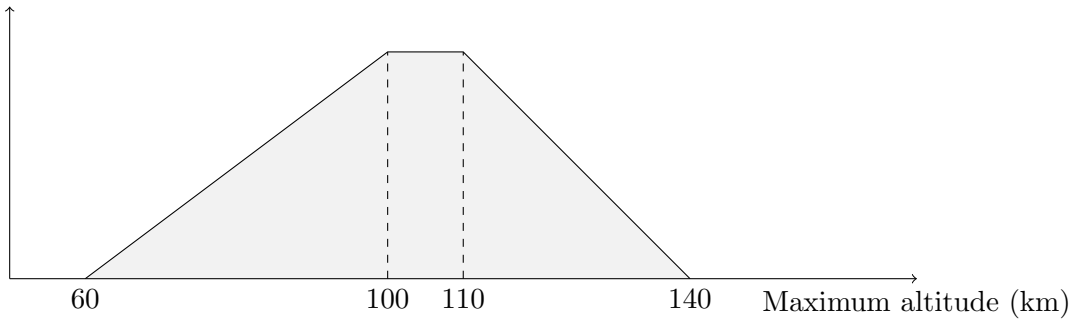



**Figure 50:** Membership function of the maximum altitude constraint

#### *4.2.1.2 Maximum Load Factor*

While the seat pitch directly represents the passenger experience, there is also a need for taking into account the maximum load factor. Indeed, this factor impacts both the safety and the comfort of the passengers. According to Starke et al. [418], it is one of the main requirements for suborbital flights. They also set a limit for the maximum load

factor equal to 5 in order to limit the risk of gray-out and especially black-out. Moreover, in order to improve the passenger experience, some concepts might benefit from a lower maximum load factor, such as the RocketPlane XP, which only reaches 3.5. Based on this analysis, the uncertainty of the maximum load factor constraint is modeled with a trapezoidal membership function, as displayed in Figure 51.

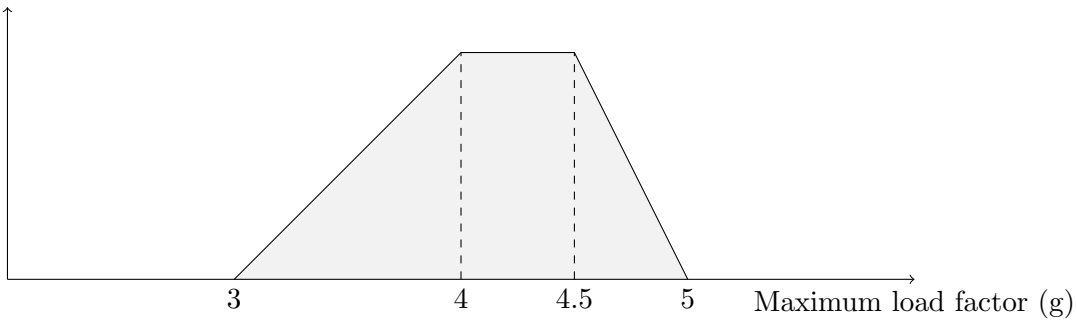



**Figure 51:** Membership function of the maximum load factor constraint

#### *4.2.1.3 Yearly Number of Passengers*

Designing the best vehicle for the future suborbital market requires designers to model the potential demand. Since the market is new, the amount of uncertainty is relatively large. To account for this uncertainty, the Futron Corporation developed three demand scenarios based on different market maturity dates [439]. The baseline scenario forecasts 15,000 passengers per year against 11,000 for the pessimistic and 30,000 for the optimistic, as displayed in Figure 52.

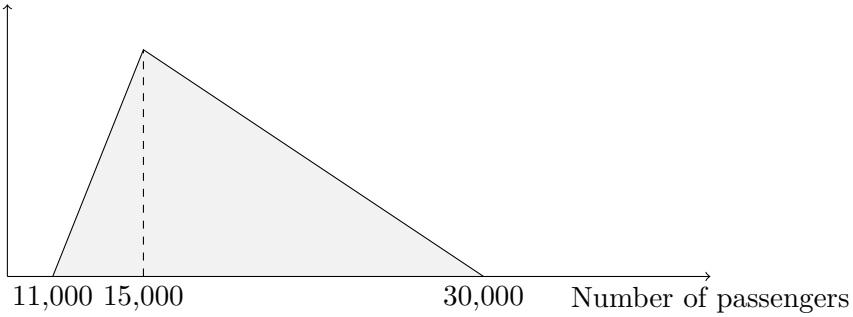



**Figure 52:** Membership function of the yearly number of passengers constraint

### 4.2.2 Time-Dependence Models

In order to model the impact of time on the various requirements, several traditional models can be applied to the scaling parameters used to represent the aforementioned membership functions. In particular, two main parameters of interest might be the mean value and the standard deviation. While the standard deviation tends to decrease over time, the mean value might follow different trends. This section provides examples of models that can be used to model time dependence of the various factors around the baseline value $\lambda_0$:

- Exponential behavior: this model is particularly useful to model a decrease in the degree of uncertainty. Indeed, it can describe a rapid decrease in the first years/months followed by a period of asymptotic and slow decrease towards zero. This scaling factor $\lambda_e$ is driven by a single parameter $t_{1/2}$ called the half-life time, as described in Equation 63. In this model, the larger the half-life time, the slower the decrease.

$$\lambda_e(t) = \lambda_0 \exp^{\left(-t\frac{\ln 2}{t_{1/2}}\right)} \tag{63}$$

- Quadratic behavior: this model can be used to model the variation of a mean value over time that follows a quadratic trend. As described in Equation 64, the scaling factor $\lambda_q$ is driven by two parameters $a$ and $b$ that must be defined by the users.

$$\lambda_q(t) = \lambda_0 + at + bt^2 \tag{64}$$

- Linear behavior: this model can be used to model a linear variation of any parameter based on a single constant $c$, as described in Equation 65.

$$\lambda_l(t) = \lambda_0 + ct \tag{65}$$

# CHAPTER V

# STEP 2: DEFINE THE DESIGN SPACE

This step aims at determining all feasible alternatives that will be considered in the design space exploration. As discussed in Section 2.1, a modified version of the morphological matrix is developed in order to improve the efficiency of the overall optimization process. This new method is implemented into a software called Efficient Variable-Oriented Software for Architecture Generation (ENVISAGE), whose its development is a part of this work.

## *5.1 Improvement of the Morphological Matrix*

The objective of the new method is to generate all feasible alternatives in a way that they can be efficiently optimized. While the typical morphological matrix provides a powerful method to generate all possible alternatives, it does not ensure the feasibility of the obtained alternatives. To overcome this drawback, the morphological matrix is combined with a compatibility matrix. While this combination is able to provide all feasible alternatives, it does not account for design variables. In addition, since a single optimizer can only handle alternatives that are defined by the same design variables, there is a need for grouping them. Hence, alternatives that are defined by the same design variables are grouped into a single architecture. For instance, straight wings, delta wings, and swept wings are grouped into a single wing category and are defined by their sweep angle, surface area, aspect ratio, etc. This allows for each generated architecture to be optimized by a single optimizer. The output of this method is a list of all feasible architectures and their corresponding design variables. The development of the aforementioned method can be decomposed into four steps:

1. Generate alternatives: alternatives are generated using a morphological analysis. This method is particularly useful for multi-dimensional complex problems since it provides a structured, functional, and intelligent way to decompose a problem and generate

alternatives [340]. One of the most common practices in morphological analysis is the use of a morphological matrix, or matrix of alternatives. This matrix is a two-dimensional representation of the system. Each row represents a function/feature of the system and each column represents an option. The number of possible alternatives is given by the product of the number of options of all functions. If the morphological matrix $M$ is defined by its rows $i$ and its columns $j$, the number of alternatives is defined in Equation 66.

$$\mathrm{N_{alt}} = \prod_i \left( \sum_j M^*_{i,j} \right) \quad \text{where} \begin{cases} M^*_{i,j} = 1 & \text{if } M_{i,j} \neq \emptyset \\ M^*_{i,j} = 0 & \text{if } M_{i,j} = \emptyset \end{cases} \tag{66}$$

2. Ensure feasibility: feasibility is ensured by only selecting the combinations of options for which all options are compatible with each other. A square matrix called compatibility matrix is used to define compatibility between options. This matrix $C$ is symmetric so that only the upper (or lower) triangular needs to be completed by either 0 or 1 according to the rule presented in Equation 67. In this equation, $k$ and $l$ represent the $k^{\text{th}}$ row and the $l^{\text{th}}$ column of the matrix.

$$C_{k,l} = \begin{cases} 1 & \text{if options } k \text{ and } l \text{ are compatible} \\ 0 & \text{if options } k \text{ and } l \text{ are incompatible} \end{cases} \tag{67}$$

Using the definition of the compatibility matrix, the diagonal can be prefilled by ones since a given option is necessarily compatible with itself. In addition, while functions can be described by different options, only one option per function can be selected for a given design alternative. As a consequence, all options used to define the same function are necessarily incompatible. Therefore, additional entries can be prefilled. As a consequence, the final number of entries that must be filled by designers is reduced to $N_f$, as defined in Equation 68.

$$N_f = \frac{N_{alt}\left(N_{alt}-1\right)}{2} - \sum_i \frac{\left(\sum_j M^*_{i,j}\right)\left(\left(\sum_j M^*_{i,j}\right)-1\right)}{2} \tag{68}$$

3. Assign variables: the design variables need to be identified to enable alternatives to be grouped into architectures. First, all design variables that are used by at least one option of a specific function are defined. Then, the listed variables are assigned to options using assignment matrices. These matrices are composed of checkboxes to link variables to options.

4. Generate feasible architectures: based on both the morphological matrix and the compatibility matrix, an algorithm will first determine the list of feasible alternatives. Then, based on this list and the assigned design variables, alternatives that are described by the same variables are grouped into architectures.

This methodology has been implemented into a new software written in Matlab called ENVISAGE. Its development is described in the following section.

## *5.2 Development of the EfficieNt Variable-orIented Software for Architecture GEneration (ENVISAGE)*

The general structure of the tool is based on the method described in the previous section. First, the users input their morphological matrix based on a functional decomposition. Then, to ensure the feasibility of the generated concepts, they complete a pre-filled compatibility matrix. After this point, two analysis options are proposed:

- Alternative-based analysis: users can generate the list of all feasible alternatives.
- Architecture-based analysis: after defining the design variables specific to each function and option, users can generate the list of all feasible architectures.

While the general architecture of the tool is displayed in Figure 53, this section discusses the main functions of the software, further detailed in Appendix A.1.

First, users have to define all possible options using the parametric morphological matrix proposed when defining possible options. Assuming that the physical decomposition has already been performed, users list all identified functions in rows and the corresponding potential options for each function in columns. Once the morphological matrix has been fully defined, the compatibility between each couple of options must be defined by the users

using the automatically generated square compatibility matrix. Since the compatibility matrix is symmetric, only the upper triangular is considered. In addition, the compatibility matrix is automatically pre-filled to account for the fact that two options of the same feature are necessarily incompatible. Users can load a predefined matrix and save modifications of this matrix. Once the compatibility has been defined, two different analyses are proposed. The first one corresponds to the traditional approach and lists all feasible alternatives. The second one allows users to define design variables and generate architectures based on the new approach.

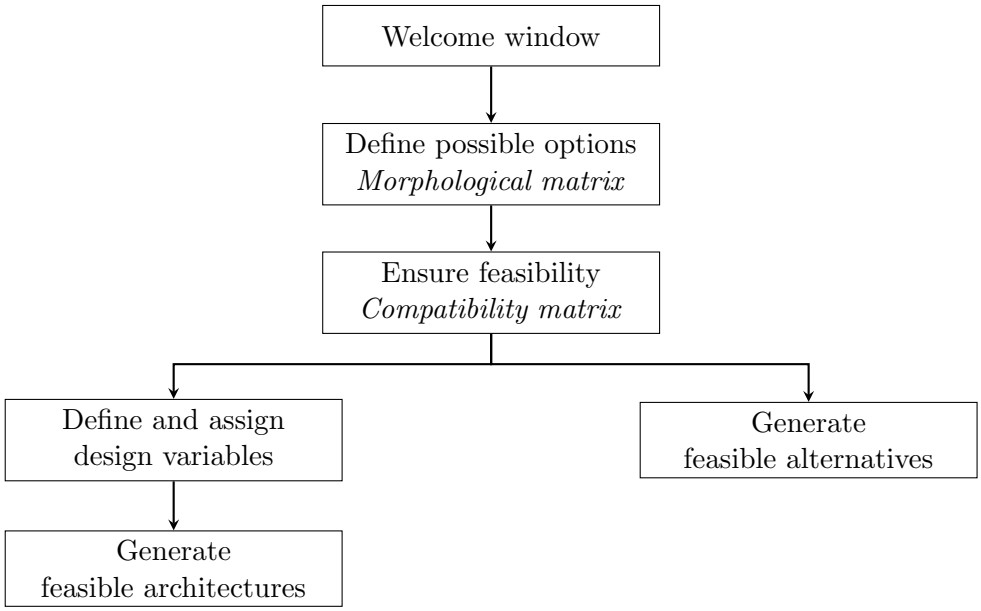



**Figure 53:** General architecture of ENVISAGE

### 5.2.1 Generate Feasible Alternatives

The list of feasible alternatives is generated based on both the morphological matrix and the compatibility matrix. The developed function converts the morphological matrix into a list of alternatives using a recursive function that searches for feasible combinations of options. This recursive function also accounts for the compatibility between an option and all other options from a given list.

### 5.2.2 Generate Feasible Architectures

In order to define the elements required to generate architectures, the users need to define the design variables that must be considered for each function and allocate them to the corresponding options. Using the variables defined for each option and the list of feasible alternatives, a set of variables is associated to each alternative. A final loop is then used to group all alternatives that are described by the same design variables into architectures.

This architecture generation process is summarized in Figure 54 and its detailed implementation into a generic user-friendly environment is described in Appendix A.1.

## *5.3 Application to Suborbital Vehicles*

The previous software is used to generate the different architectures of suborbital vehicles, which is used as Experiment 1.1. A functional decomposition of the vehicle highlights the main features of suborbital vehicles. The main objective of commercial suborbital vehicles is to carry passengers to an altitude higher than 100 km. To reach the targeted altitude, the vehicle must take-off (or be launched), fly, and land safely. To be able to fly, the vehicle must create lift and/or thrust. Since the vehicle must fly at very high velocity and high altitude, a rocket engine is needed. In addition, it can be seconded by different types of jet engines. The vehicle must also be laterally and longitudinally stable and controllable in the atmosphere and in space. The functional decomposition is displayed in Figure 55. The gray boxes correspond to the end functions that will appear in the morphological matrix while the white boxes correspond to high-level features that are further decomposed.

Based on the previously defined functions, the options are generated based on existing concepts. More than 30 vehicles have been investigated and decomposed in order to find the possible options [22, 252, 268, 484]: SpaceJet (Astrium), New Shepard (Blue Origin), Vehra (Dassault Aviation), Soar (Swiss Space Systems), SpaceShipTwo (Scaled Composites), Lynx 2 (X-COR), Space Cruiser (Vela Technology Development), AS&T Suborbital Aerospaceplane (Andrews Space & Technology), Lucky Seven (Acceleration Engineering), Advent

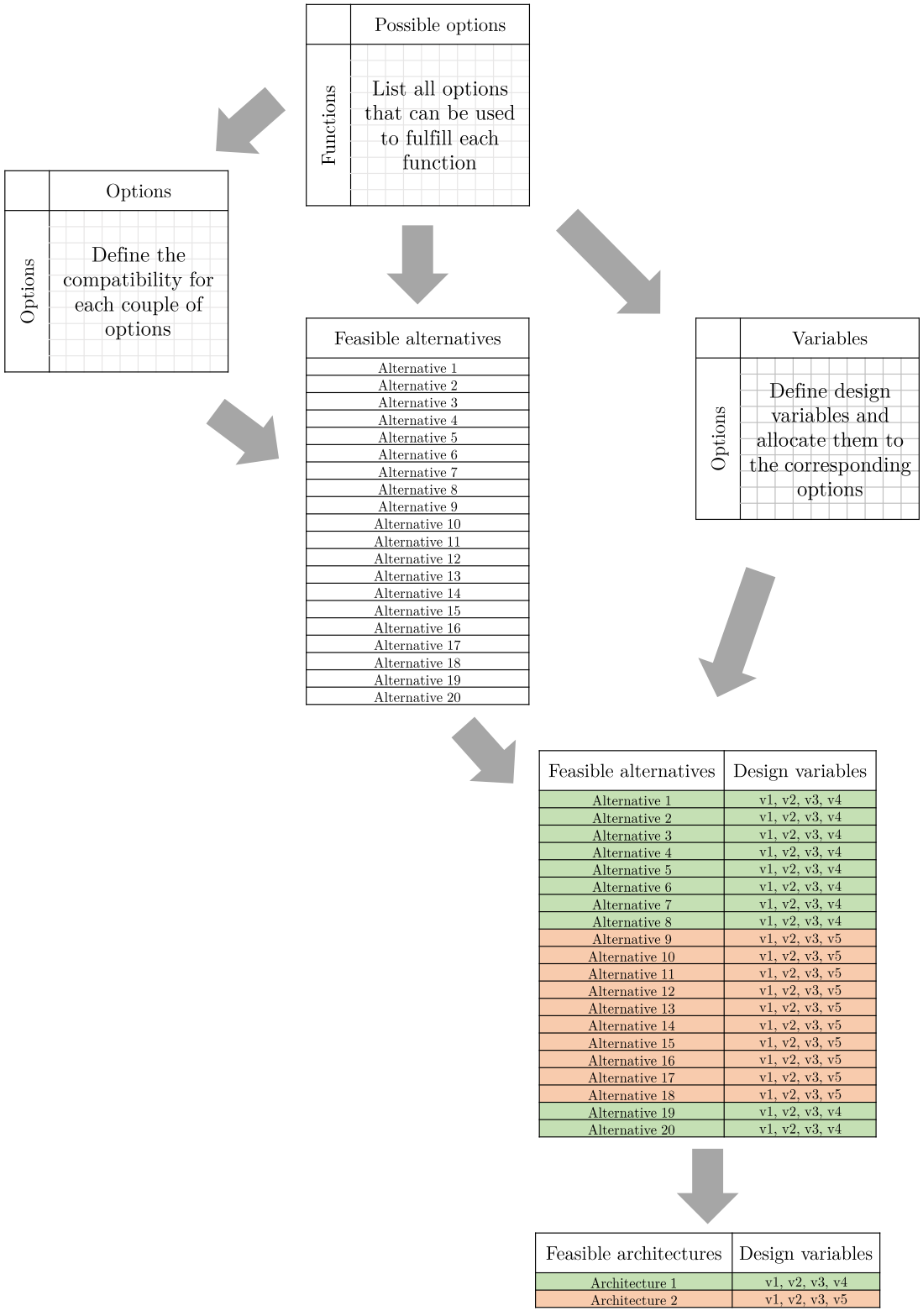



**Figure 54:** Architecture generation process

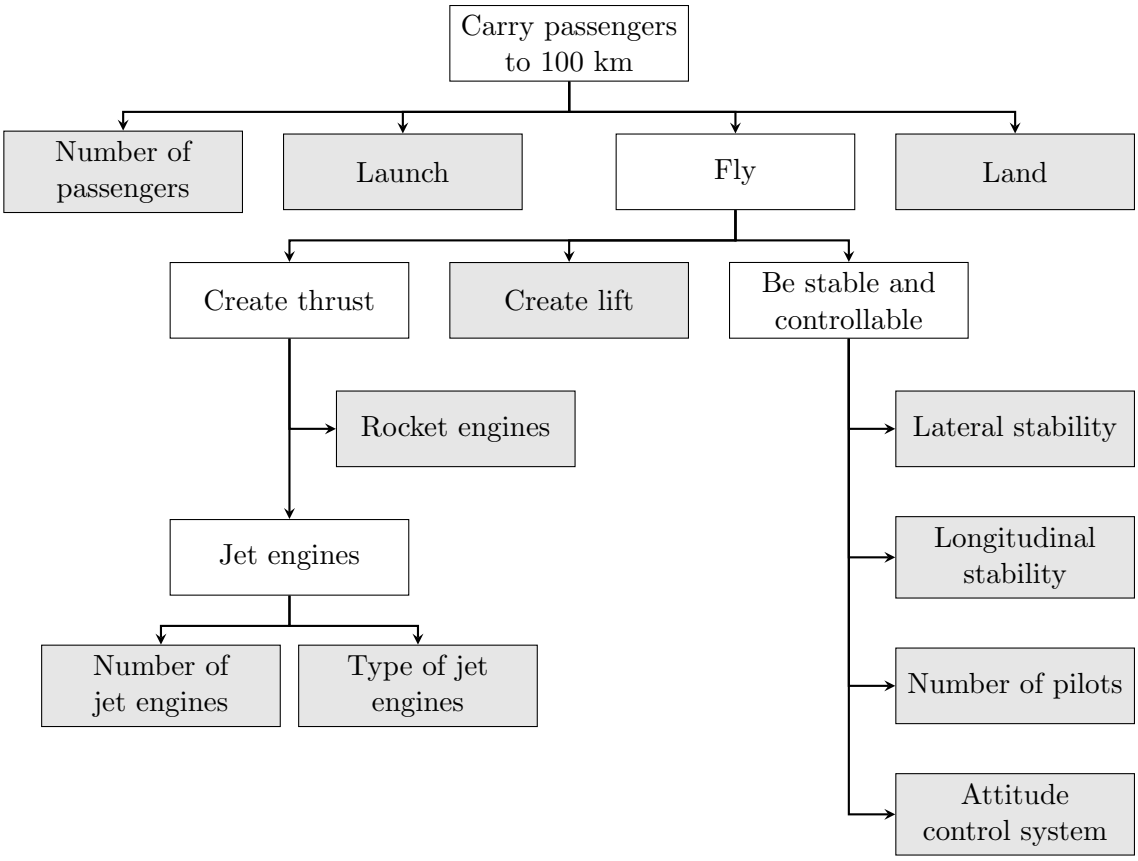



**Figure 55:** Functional decomposition of suborbital vehicles

(Advent Launch Services), ORIZONT (Aeronautics and Cosmonautics Romanian Association), Black Armadillo (Armadillo Aerospace), The Spirit of Liberty (American Astronautics Corporation), Ascender (Bristol Spaceplanes, Ltd), Canadian Arrow (Canadian Arrow), Wild Fire (The da Vinci Project), Gauchito-The Little Cowboy (Pablo de Leon & Associates), The Space Tourist (Discraft Corporation), The Green Arrow (Flight Exploration), Aurora (Fundamental Technology Systems), Liberator (HARC), Negev 5 (IL Aerospace Technologies), Solaris X (Interorbital Systems), Astroliner (Kelly Space and Technology), Cosmos Mariner (Lone Star Space Access Corporation), Crusader X (Micro-Space, Inc.), SabreRocket (PanAero, Inc.), XP (Pioneer Rocketplane), SpaceShipOne (Scaled Composites), Thunderbird (Starchaser Industries), Cosmopolis XXI (Suborbital Corporation), MICHELLE-B (TGV Rockets), and Eagle (Vanguard Spacecraft).

Using the information about the aforementioned concepts, options for each of the identified functions can be found to provide the required data to build the morphological matrix related to suborbital vehicles (Table 22):

- Type of launch: suborbital vehicles can take off horizontally (similar to typical aircraft) or vertically (similar to rockets). In addition, some concepts have been launched by an intermediate vehicle: either an aircraft (SpaceShipOne) or a balloon (Wild Fire).

- Type of landing: suborbital vehicles can land horizontally with power (similar to typical aircraft) or without power (similar to the SpaceShipOne and the Space Shuttle). In addition, some concepts can also land using retro-rockets (New Shepard) or a parachute (Black Armadillo) to slow down their rate of descent.

- Lift generation: both slender bodies and winged bodies have been used for existing concepts. Slender bodies do not generate lift and only rely on their thrust to fly (Black Armadillo). Winged bodies can be equipped with straight wings (SpaceJet), delta wings (Vehra) or swept wings (Rocketplane XP).

- Longitudinal stability: for vehicles equipped with wings, there might be a need for a second horizontal lifting surface that helps controlling the stability of the vehicle. This can be fulfilled either by an horizontal stabilizer or by a canard.

- Lateral stability: for vehicles equipped with wings, there is a need for a vertical surface that controls the stability of the vehicle. This can be fulfilled either by a vertical stabilizer or by large wing tips.

- Type of rocket engine: the three main types of chemical rocket engines can be used to power a suborbital vehicle from low altitude to around 100 km. In addition, the range of thrust used in suborbital vehicles corresponds to the trade-off zone between pressurized and pump-fed liquid engines. Therefore, they must both be considered in addition to solid and hybrid rocket engines.

- Number of jet engines: in order to benefit from an efficient propulsion at low altitude,

the rocket engine can be seconded by jet engines. If present, the number of jet engines can reach four, similar to the biggest commercial aircraft in service.

- Type of jet engines: since the jet engine has to operate during a short period of time, provide a large amount of thrust, and evolve at high speed and high altitude, only turbojet and turbofan engines are considered in this research.

- Number of pilots: while most of the concepts include human pilots (1 or 2), some of them are fully automated (New Shepard).

- Number of passengers: While all commercial suborbital vehicles aim at carrying passengers (or at least an equivalent weight of payload), the number of passengers greatly varies between concepts. Indeed, it goes from 1 (Ascender) to 8 (Space Cruise).

- Attitude control system: in order for the vehicle to be controllable for the re-entry phase, the vehicle must be equipped with an attitude control system. According to existing concepts, the latter can be either composed of cold gas engines (Canadian Arrow) or liquid rocket engines (Lynx 2).

**Table 22:** Morphological matrix for suborbital vehicles

| | **Option 1** | **Option 2** | **Option 3** | **Option 4** | **Option 5** | **Option 6** | **Option 7** | **Option 8** | **Number of alt.** |
|---|---|---|---|---|---|---|---|---|---|
| **Type of launch** | Horizontal | Vertical | Aircraft launched | Balloon launched | | | | | 4 |
| **Type of landing** | Horizontal powered | Gliding | Rocket | Parachute | | | | | 4 |
| **Lift generation** | Delta | Swept wing | Straight wing | None | | | | | 4 |
| **Longitudinal stability** | Horizontal stabilizer | Canards | None | | | | | | 3 |
| **Lateral stability** | Vertical stabilizer | Wing tip | None | | | | | | 3 |
| **Type of rocket engine** | Pressurized liquid | Pump-fed liquid | Solid | Hybrid | | | | | 4 |
| **Number of jet engines** | 0 | 1 | 2 | 3 | 4 | | | | 5 |
| **Type of jet engines** | Typical turbojet | Augmented turbojet | Typical turbofan | Augmented turbofan | None | | | | 5 |
| **Number of pilots** | 0 | 1 | 2 | | | | | | 3 |
| **Number of passengers** | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 8 |
| **Attitude control** | Cold gas | Liquid | | | | | | | 2 |
| **Number of possible combinations** | | | | | | | | | **2,764,800** |

Based on this literature review, the morphological matrix presented in Table 22 is inputted in ENVISAGE. A first raw alternatives generation would result in a total of 2,764,800 possible combinations. However, among these combinations, some are incompatible. Hence, the next step aims at removing all incompatible combinations from the generated list.

The compatibility between options is set based on the authors' knowledge and judgment. For example, a slender body cannot take off or land horizontally. In addition, there is no incentive to carry jet engines if the vehicle is launched from an aircraft or from a balloon. Other similar considerations are made to build the compatibility matrix. Once completed, ENVISAGE is executed in order to extract all feasible alternatives. A list of 123,000 feasible alternatives is provided.

### 5.3.1 Variable Definition and Assignment

To create architectures, the design variables used to describe each function have to be listed and assigned to their specific options. The list of all design variables considered in this study is provided below:

- Type of launch: initial speed $V_i$, initial altitude $h_i$, take-off field length $L_{TO}$
- Type of landing: landing field length $L_{LA}$, landing speed $V_{LA}$, engine re-start altitude $t_r$
- Lift generation: reference area $S_{ref}$, sweep angle $\Lambda$, aspect ratio $AR$
- Longitudinal stability: horizontal surface area $S_h$, maximum thickness-to-chord ratio of the horizontal surface $t_{c,h}$
- Lateral stability: vertical surface area $S_v$, maximum thickness-to-chord ratio of the vertical surface $t_{c,v}$
- Type of rocket engines: propellant $prop$, nozzle expansion ratio $\epsilon$, chamber pressure $p_c$, thrust $T_r$, burning time $t_b$
- Number of jet engines: number of jet engines $n_j$

- Type of jet engines: afterburner $ab$, turbine inlet temperature $T_4$, thrust $T_j$, transition altitude $h_t$
- Number of pilots: number of pilots $n_p$
- Number of passengers: number of passengers $n_{PAX}$
- Attitude control system: thrust $T_{AC}$, number of engines $n_{AC}$, burning time $t_{AC}$

### 5.3.2 Results

Once the design variables are listed and assigned to the corresponding options, ENVISAGE is executed and identifies 4 feasible architectures. A summary of the architecture generation process is provided in Figure 56. Starting with about 2.8 million total combinations, the integration of the compatibility matrix within the process allows designers to divide the number of alternatives to be investigated by a factor of 23. Grouping all alternatives defined by the same design variables into architectures further enables to reduce the number of architectures to be optimized to four. Doing so allows for the number of discrete algorithms to be set to be reduced by a factor of 700,000.

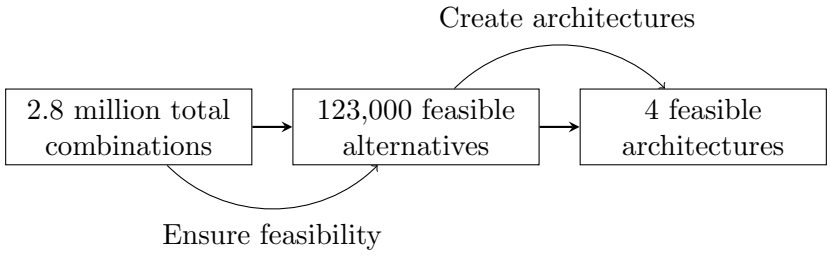



**Figure 56:** Summary of the architecture generation process

The application of the proposed approach on suborbital vehicles shows that all suborbital vehicles can be grouped into four architectures, as described below and illustrated in Figure 57:

- Architecture 1: slender vehicles that take off vertically without jet engines (New Shepard).

- Architecture 2: winged vehicles that take off from the ground horizontally or vertically without jet engines (Lynx II).
- Architecture 3: winged vehicles that take off from the ground with jet engines (Rocketplane XP).
- Architecture 4: winged vehicles without jet engines launched from an aircraft (SpaceShipTwo).

(a) Architecture 1 [40]

(b) Architecture 2 [150]

(c) Architecture 3 [108]

(d) Architecture 4 [460]

**Figure 57:** Example of each architecture

The next section discusses and quantifies in more detail the general benefits of implementing a variable-oriented morphological analysis for a complex design problem.

## *5.4 General Benefits of the New Architecture Generation Methodology*

As discussed in Section 2.1, a design space exploration usually requires complex optimization processes. The efficiency of such exploration is driven by the ability of the designers

to both cover the maximum number of concepts and reduce the execution time to reach the best concept(s). The use of morphological analysis already ensures the exhaustiveness of the concept generation process. The proposed approach focuses on reducing the execution time of the optimization process. The required number of function calls to reach the optimum value is used to better quantify the benefits of implementing a variable-oriented morphological analysis. Chelouah et al. [79] calculated the average number of function calls required for a genetic algorithm to optimize different test functions. Among these test functions, the Rosenbrock function $R_n$ (Equation 69) and the Zakharov function $Z_n$ (Equation 70) have $n$ design variables and can be used to analyze the behavior of the optimization algorithm with respect to $n$. Since they can easily be scaled to $n$ dimensions, these two functions provide a good way to measure the impact of increasing the number of design variables on the required number of function calls. In addition, they have been tested with the same convergence criteria so that the results are consistent.

$$R_n(x) = \sum_{j=1}^{n-1} \left[ 100 \left( x_j^2 - x_{j+1} \right)^2 + (x_j - 1)^2 \right] \tag{69}$$

$$Z_n(x) = \left( \sum_{j=1}^{n} x_j^2 \right) + \left( \sum_{j=1}^{n} 0.5 j x_j \right)^2 + \left( \sum_{j=1}^{n} 0.5 j x_j \right)^4 \tag{70}$$

Their results for different values of $n$ are presented in Table 23. Since the behavior of the problem's specific objective function is unknown, the average number of function calls is calculated for each $n$. It is then approximated by $f_c(n) = 7.7018n^2 + 1189.5n - 1323.6$ with an $R^2$ of 0.9995.

**Table 23:** Number of function calls for $R_n$ and $Z_n$ functions [79]

| **Functions** | **Number of design variables** | | | | |
|---|---|---|---|---|---|
| | **2** | **5** | **10** | **50** | **100** |
| $R_n$ | 960 | 3,990 | 21,563 | 78,356 | 194,302 |
| $Z_n$ | 620 | 1,350 | 6,991 | 75,201 | 195,246 |
| Average | 790 | 2,670 | 14,277 | 76,779 | 194,774 |

Even though this relationship highly depends on the problem's objective function and

constraints, it is assumed that the trend in the number of function calls with respect to the number of variables remains similar to the aforementioned functions. The number of function calls required can now be compared assuming two design processes that can cover the entire design space: the first one optimizes all 2.8 million alternatives and the second one optimizes all four architectures generated by ENVISAGE. Figure 58 represents the computational time required to find the optimum value with respect to the number of design variables considered in the conceptual design of suborbital vehicles. It assumes an execution time of the design framework of one second. Using the traditional design process, the number of function calls $N_t$ is defined by $N_t = 2.8 \times 10^6 \times f_c(n)$. Using ENVISAGE, $N_t = 4 \times f_c(n)$. As illustrated in Figure 58, implementing the aforementioned variable-oriented morphological analysis helps significantly reduce the required execution time, hence supporting the complete exploration of large design spaces.

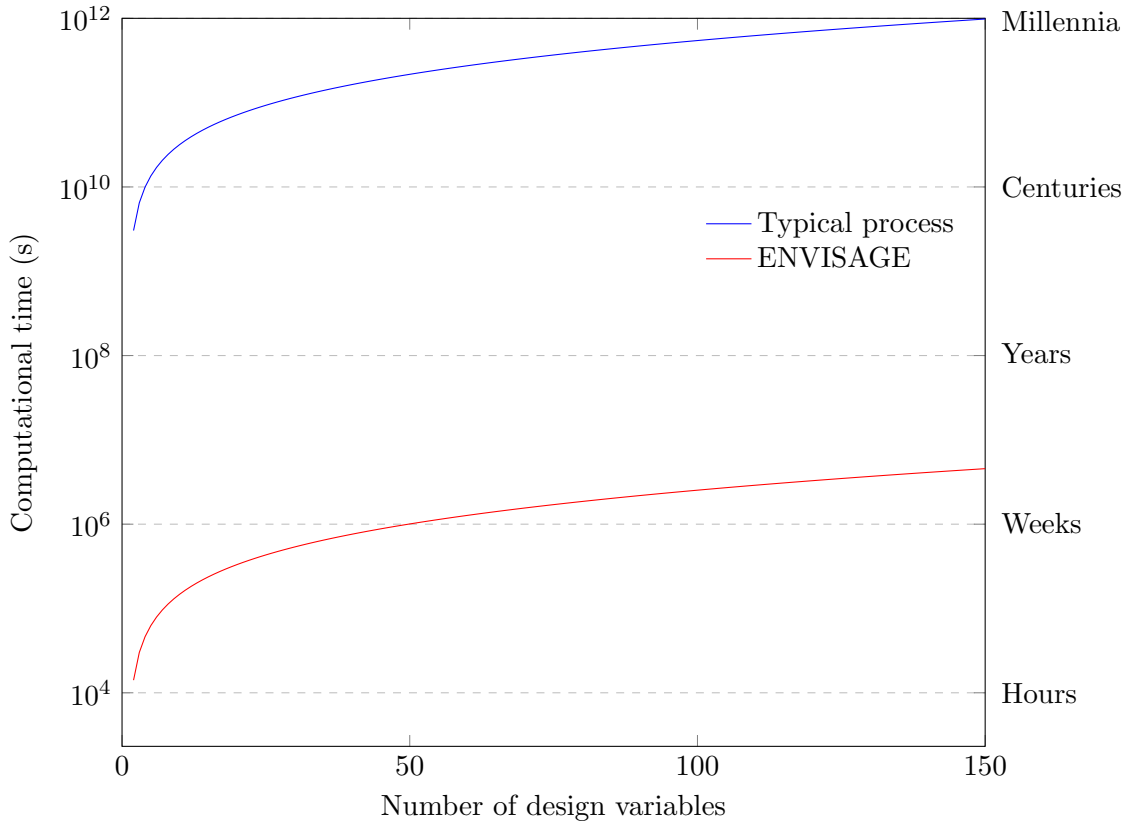



**Figure 58:** Improvements in computational time

In order to estimate the benefit of grouping options against the disadvantage of adding more design variables, the factor $k$ is introduced. $k$ represents the number of variables that must be added to the optimization process when grouping two options. For instance, if $k = 5$, removing one option from a function requires five more variables in the optimization process. To perform trade-off analyses, a baseline morphological matrix is used, which is composed of 15 functions and 8 options per function, initially defined by 10 variables. Hence, the total number of combinations is $8^{15} = 3.5 \times 10^{13}$, which initially requires $8^{15} f_c(150) = 1.2 \times 10^{19}$ function calls to be completely investigated. Figure 59 displays the improvement in the number of function calls with respect to the number of options removed per function.

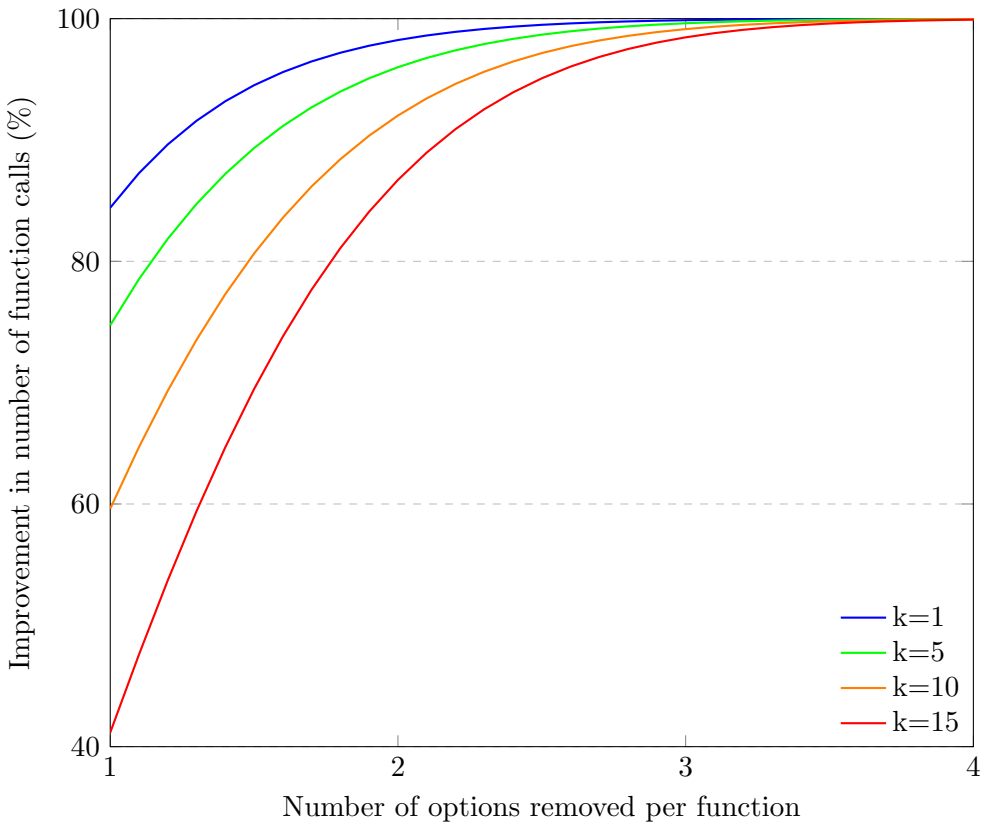



**Figure 59:** Improvements in the number of function calls

As expected, the benefits are smaller as the number of variables added per function removed increases. The results also demonstrate that, even for 15 variables added per option removed, the improvement exceeds 40%. In addition, grouping 3 out of the initial 8

options results in large improvements, independently of the number of variables added.

A sensitivity analysis is conducted to assess the sensitivity of improvement to changes in the number of functions, number of variables per function, etc. A DoE is generated using the statistical software JMP®. A Latin Hypercube is used to cover the entire design space with 3,000 points. The variables and their corresponding ranges are presented below:

- Number of functions: $[10; 30]$
- Number of options per function: $[5; 15]$
- Number of variables per function: $[2; 10]$
- Number of variables added per option removed: $[1; 15]$
- Number of functions removed per function: $[1; 5]$

The results are presented in Figure 60. Each curve represents the sensitivity of the overall improvement in the number of function calls with respect to changes in a specific parameter. For a given curve, this analysis assumes that all other parameters are constant. Therefore, each curve corresponds to partial derivatives of the improvement in the number of function calls with respect to a specific parameter. As shown in Figure 60, the amplitude of the improvement in the number of function calls is mainly driven by the number of options removed per function and the number of functions considered in the morphological matrix.

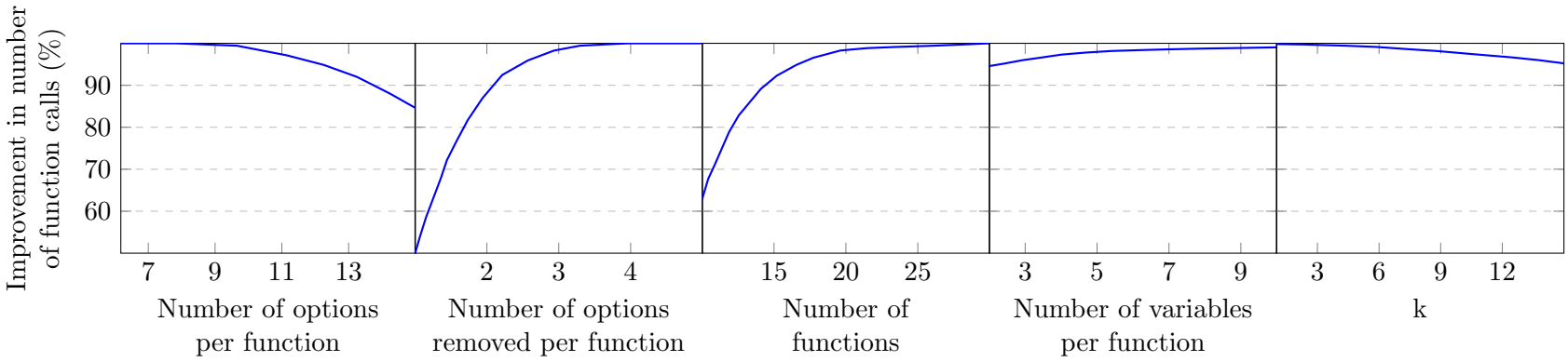



**Figure 60:** Sensitivity of the overall improvement with respect to each parameter

This analysis demonstrates that the improvements in the number of function calls are more significant when the number of functions considered increases. In particular, it shows that the benefits of using a variable-oriented morphological analysis are more significant when dealing with larger problems that have a more detailed decomposition along with a higher number of design variables. These attributes characterize the design of complex concepts such as suborbital vehicles. The benefits of such morphological analysis decrease with the number of options per function and $k$, however their sensitivities are relatively small compared to the two key drivers.

The results of Experiment 1.1 presented in this chapter are provided below:

1. The 30 existing concepts studied in the literature are included in the list of possible alternatives generated by the proposed methodology.

2. The integration of the compatibility matrix within the process constrains all the generated alternatives to be feasible.

3. The number of discrete optimization problems to be executed is reduced by a factor of $5 \times 10^5$.

4. Each architecture is defined by a unique set of design variables and can consequently be further optimized by a single optimization algorithm.

Hence, these observations result in the validation of Hypothesis 1.1:

**VALIDATION HYPOTHESIS 1.1: IF a variable-oriented morphological analysis combined with a compatibility analysis is developed THEN all feasible alternatives can be systematically generated for further comparison and optimization.**

This step presented a new method for the generation of variable-oriented architectures in a way that they can be efficiently optimized and compared by an optimization process. The next step will present the design framework required to evaluate the multi-disciplinary performance of each architecture.

# CHAPTER VI

# STEP 3: EVALUATE ALTERNATIVES

This chapter discusses the development of a design framework capable of evaluating each concept generated in the previous chapter in terms of performance, life-cycle costs, and safety. The complexity of such vehicles opens up an immense design space that must be investigated by the designers to ensure that the best concept is selected. Such exploration is based on a multi-disciplinary multi-objective optimization process, which usually requires a large number of function calls. Hence, as discussed in Section 3.3, the design framework must ideally benefit from the following capabilities:

- Conceptual design level: the environment needs to include design variables commonly used at the conceptual level.
- Physics-based modeling: in addition to the architecture selection, the environment must enable the optimization of a given architecture and the corresponding trajectory. These capabilities implicitly require the use of physics-based modeling techniques in addition or in place of historical data.
- Integration: the different modules must be integrated within a single environment, which can be further integrated into a complex optimization environment.
- Automation: the environment must be able to be implemented within an automatic loop that executes several hundreds of thousands runs to cover the entire design space and consider requirements' uncertainty. The environment must also be fast to run.

As detailed in Section 3.3, the design framework is composed of six modules embedded into a modified MDF structure, as presented in Figure 61.

First, this chapter describes the atmospheric model chosen for this research, as it is crucial for an accurate estimation of most of the aerodynamic and propulsive parameters.

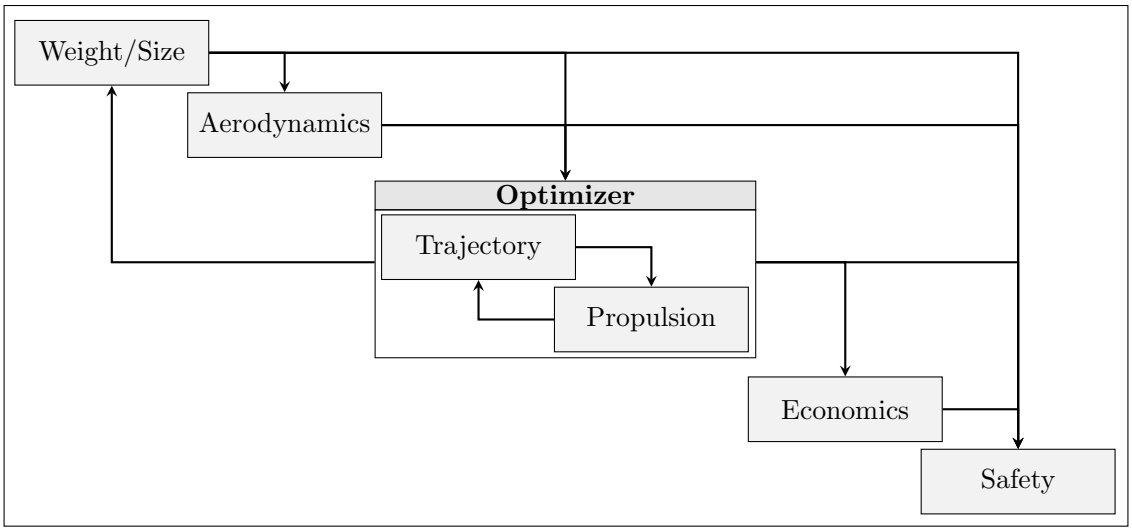



**Figure 61:** Structure of the design framework

Then, the different modules presented in Figure 61 are developed and individually validated. Finally, the overall framework is validated using existing concepts.

## *6.1 Atmospheric Model*

In order to accurately compute aerodynamic coefficients and simulate the vehicle's trajectory, atmospheric parameters must be modeled. Figure 62 displays the different atmospheric layers that can be identified due to differences in their characteristics. Suborbital vehicles maneuver in a range of altitudes between 0 and 100 km. Therefore, all atmospheric parameters such as pressure, temperature, air density, Earth gravity, speed of sound, and dynamic viscosity have to be modeled over this entire range. Using an existing atmospheric model, a dedicated Matlab function was developed to provide an easy access to the required atmospheric parameters in the main program. This section details the models used for each atmospheric characteristic. The models used to determine the different parameters are detailed in Appendix B.1.

The following section discusses one of the key modules of the design framework, which is the weight and size estimation.

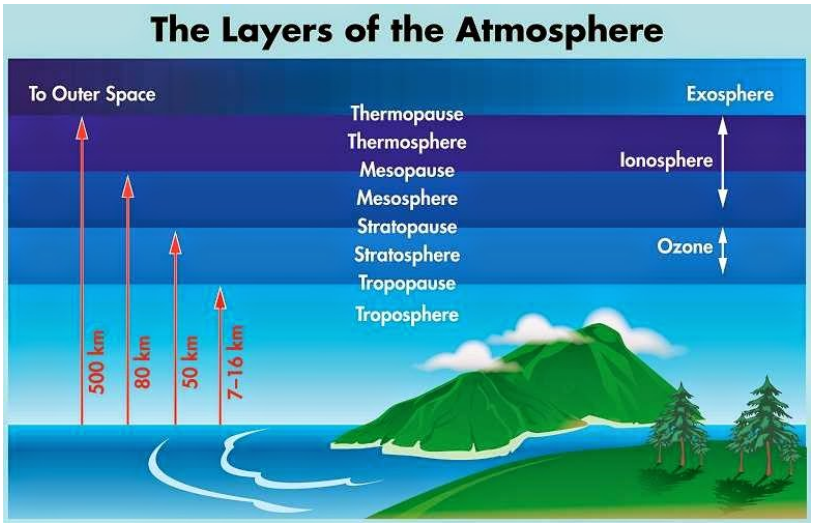



**Figure 62:** Atmospheric layers [74]

## *6.2 Weight and Size Modeling*

This section aims at describing the weight/size module of the design framework. As discussed in Section 3.3.2, the vehicle has to be physically decomposed into components so that their weight can be individually estimated.

### 6.2.1 Methodology

The selection of the weight estimation models for each component is based on the evaluation and comparison performed by Rohrschneider [366]. The presented approach is based on a physical decomposition of the vehicle into 19 subsystems: fuselage, thrust structure, nose, Thermal Protection System (TPS), wing, landing gear, horizontal tail, vertical tail, hydraulics, parachute and retrorockets, Reaction Control System (RCS), avionics, Environmental Control and Life Support System (ECLSS), primary power system, flight control, electrical systems, seats and accessories, parachute, and unused liquids. As discussed in Section 3.3.2, to account for solid, liquid, and hybrid engines, a specific module is developed for the weight estimation of the following components: the rocket engine propulsion system, the rocket engine propellants, and the tanks for rocket engine propellants. This module will be developed in Section 6.3. The latter also describes the weight estimation approach for jet engines. Due to interdependence effects, a loop must be developed in order

to generate a consistent design, as presented in Figure 63. The convergence is performed on the take-off gross weight of the vehicle. The overall methodology as well as the equations used for each component are presented in this section.

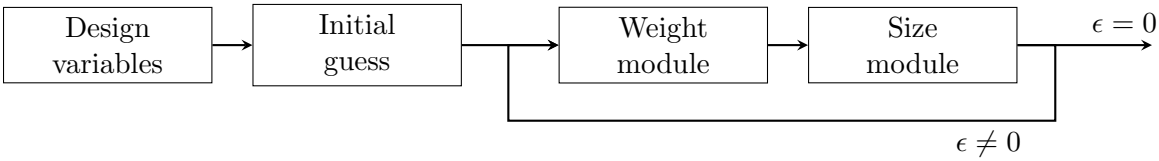



**Figure 63:** Weight/size estimation process

### 6.2.2 Description of the Weight Estimation Models

The body weight, which includes the fuselage, the thrust structure, and the nose, is estimated using Brothers' research [55]. The weight estimation of the external insulation is based on Brady et al. [47] who propose an empirical model. The wing, landing gear, hydraulic system, RCS, ECLSS, primary power system, electrical system, seat, flight control system, horizontal tail, and vertical tail weight estimation models were derived by MacConochie and Klich using aircraft data and the Space Shuttle [266]. The avionic weight can be estimated with the empirical approach suggested by Raymer [354]. As the weight of the parachute system is hard to estimate, historical data from both the Orion Multi-Purpose Crew Vehicle and the Soyuz are used as a reference [85, 192]. Finally, the weight of the unused propellant is estimated based on multiple studies [47, 55, 266, 425].

Since most of the aforementioned models have been developed between 1985 and 2000, there is a need to take into account recent enhancement in materials. For that purpose, Talay proposes a list of TRF that would adjust the weight of each component [425].

All the models used for the weight estimation module are further detailed in Appendix B.2.

### 6.2.3 Description of the Size Estimation Model

The size estimation model aims at determining the size of two main types of components: the fuselage and the lifting/control surfaces. The different models used in this research are

described in this section along with the corresponding assumptions.

#### *6.2.3.1 Lifting and Control Surfaces*

If present, the size of the lifting and control surfaces is driven by aerodynamic constraints, especially during take-off and landing. This section discusses the independent design variables that will be changed during the design process and their relationships to other parameters.

All surfaces are assumed to be trapezoidal and are defined by five independent design variables: surface area $S$, aspect ration $AR$, taper ratio $TR$, sweep angle $\Lambda$, and thickness-to-chord ratio $t_c$. Figure 64 shows the geometric variables used to define the wing. These geometric variables can be calculated using the five variables listed above.

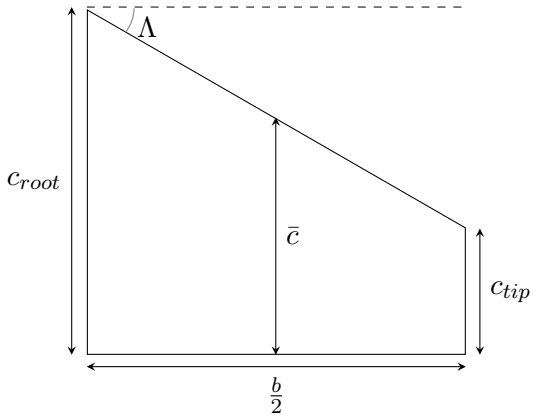



**Figure 64:** Geometry of a half lifting/control surface

The wing span $b$ can be calculated using Equation 71.

$$AR = \frac{b^2}{S} \tag{71}$$

The root chord $c_{root}$, which is also related to the tip chord $c_{tip}$ using $c_{tip} = TR\, c_{root}$, can be determined using Equation 72.

$$c_{root} = \frac{2S}{b\,(1 + TR)} \tag{72}$$

Finally, the mean aerodynamic chord $\bar{c}$ can be determined using Equation 73.

$$\bar{c} = 2c_{root}\frac{1 + TR + TR^2}{3\,(1 + TR)} \tag{73}$$

#### *6.2.3.2 Fuselage*

This section aims at defining the constraints that drive the size of the fuselage in order to find the smallest design in terms of length and diameter that meets the following requirements:

- To safely and comfortably carry passengers.
- To provide a working environment to pilots.
- To carry the propellant and the fuel required for the mission.
- To provide a sufficient torque to the tail (if present).

To do so, the fuselage can be decomposed into components along the longitudinal axis: nose, cockpit, cabin, hydraulic, electrical and mechanical systems, rocket engine, and jet engine fuel tank (if present).

**Nose:** The size of the nose can be estimated using historical data from similar vehicles such as fighter aircraft. On average, typical fighter aircraft have a nose of about 50 cm [157, 375]. As a consequence, the nose of the vehicle is assumed to be an hemisphere with a radius of 50 cm.

**Cockpit:** The cockpit must be sized so that the pilot(s) can safely control and maneuver the aircraft and communicate with the ground. Pilots take the central place of the cockpit and each element must be designed to help them perform their duties. The cockpit of the vehicle is assumed to follow the same rules as the ones of fighter aircraft, as presented in Figure 65 provided by Sadraey [375]. Therefore, the cockpit is assumed to have a length of 150 cm along the longitudinal axis.

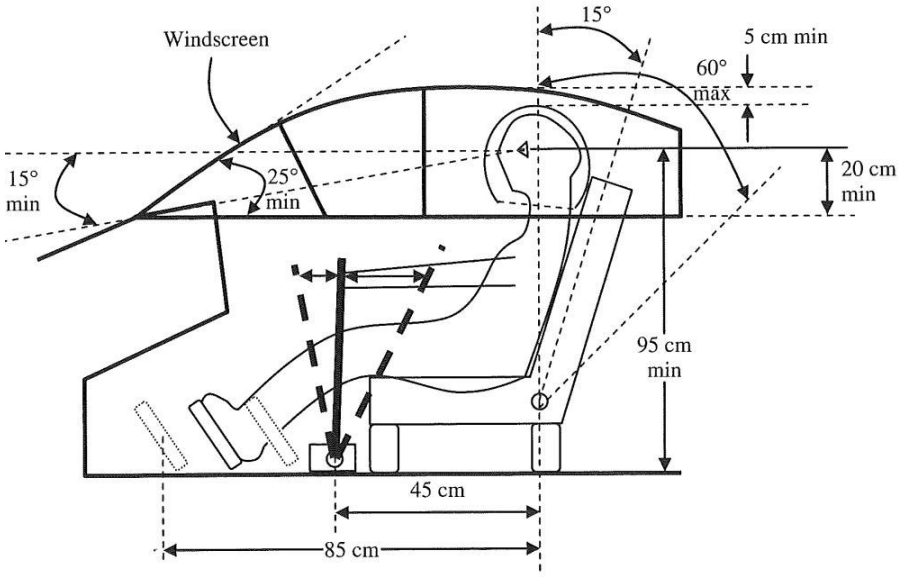



**Figure 65:** Typical cockpit design for fighter aircraft [375]

**Cabin:** For comfort purposes, a cabin configuration similar to typical commercial first class is selected. The dimensions of the seats and required spaces are provided by Sadraey and presented in Table 24 [375]. Additional legroom is added to allow passengers to enjoy the weightlessness period. Therefore, based on these data, the minimum fuselage diameter is 200 cm, while adding one row of seats increases the fuselage by 150 cm. These dimensions are very close to the ones of the SpaceShipTwo [457]. Assuming that 80% of the fuselage volume is used for the cabin in this section, it provides about 3 $m^3$ per passenger.

**Table 24:** Key cabin parameters [375]

| Parameters | Values (cm) |
|---|---|
| Seat width | 70 |
| Seat pitch | 100 |
| Aisle width | 60 |
| Cabin height | 180 |

**Hydraulic, electrical, and mechanical systems:** All hydraulic, electrical, and mechanical systems are grouped together in a single portion of the fuselage. According to

historical data from Sadraey, around 50 cm of the fuselage must be dedicated to those systems [375].

**Rocket engine:** Due to their complexity, the dimensions of the rocket engine are calculated in Section 6.3, which is dedicated to propulsion systems.

**Jet engine fuel tank:** The volume of the fuel tank is directly related to the fuel required for the mission. Assuming a cylindrical tank with the same diameter as the fuselage $d_f$, the length of this tank $L_{jt}$ can be computed using Equation 74, where $m_f$ is the fuel mass, and $\rho_f$ the fuel density.

$$L_{jt} = \frac{4m_f}{\pi d_f^2 \rho_f} \tag{74}$$

### 6.2.4 Description of the Inputs and Outputs of the Weight/Size Module

The module previously described requires 28 inputs that can be categorized into 6 different categories:

- Wing: surface, thickness-to-chord ratio, taper ratio, and aspect ratio
- Configuration: presence/absence of wing, horizontal tail, vertical tail, secondary engine, and number of pilots
- Rocket engine: vacuum thrust, type of propellant, chamber pressure, nozzle expansion ratio, combustion time, propellant weight, and engine diameter
- Jet engine: thrust at sea level, number of jet engines, bypass ratio, presence/absence of afterburner, turbine inlet temperature, and fuel weight
- Fuselage: diameter of the fuselage, length of the front fuselage, length of the rear fuselage, and diameter of the fuselage base
- Requirements: maximum load factor, number of passengers, and seat pitch

Based on these inputs, the weight module outputs all the required weights and dimensions of the vehicles and their components. The details of the implementation of the module can be found in Appendix D.2.1.

### 6.2.5 Application to Suborbital Vehicles

The proposed approach has been applied to the SpaceShipTwo for validation purposes, assuming the actual propellant weight. The calculated empty weight of the vehicle is 6,381 kg and its length is 16.5 m. The distribution of the major weight contributors is presented in Figure 66. Compared to actual values, there is a relative error of 4.2% on the empty weight and 9% on the length of the vehicle.

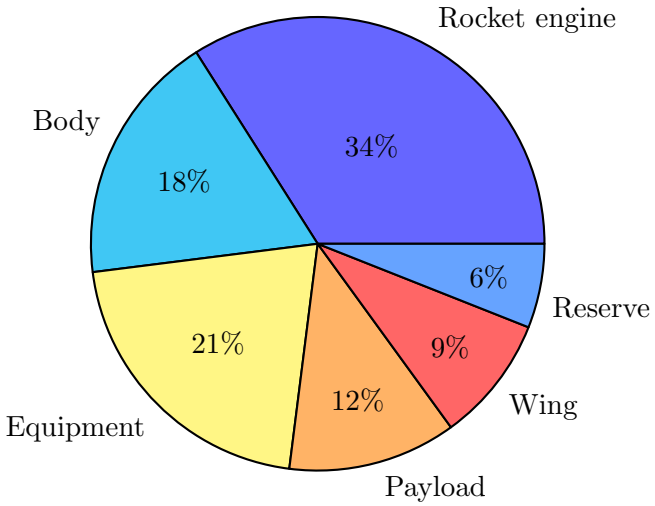



**Figure 66:** Calculated weight distribution of the SpaceShipTwo

Based on these results, the weight/size module is considered to be validated and accurate enough for conceptual level considerations.

This section presented the weight/size estimation approach for all non-propulsive subsystems. The modeling of the propulsion system, which is a key driver of the vehicle's performance, weight, and size is discussed in the following section.

## *6.3 Propulsion Modeling*

As displayed in Figure 66, the rocket engine represents a large part of the empty weight. In addition, its performance highly impacts the fuel weight, which might represent up to 75% of the vehicle take-off gross weight. As a consequence, the sensitivity of the propulsion system on the overall vehicle makes it crucial to model and requires more physics-based analysis than the other components. In addition, contrary to the weight/size and the

aerodynamic modules, there is a gap in current techniques that prevents the use of existing techniques. Hence, this section aims at precisely modeling all three types of chemical rocket engines as well as the various types of jet engines.

### 6.3.1 Rocket Engines

This section aims at developing a design framework that enables the rapid performance, weight, and size evaluation of the main propulsion system. An accurate modeling of the main propulsive system is crucial since it represents an important part of the overall vehicle characteristics. This section first describes the overall methodology. Then, the performance module is detailed and validated. Finally, the weight/size module for each type of propellant is discussed.

#### *6.3.1.1 Methodology*

Section 3.3.3 discussed the various performance, weight, and size estimation tools. In particular, it highlighted the lack of tools and methods that meet all the identified requirements for performance, weight, and size estimation. Table 25 shows that a combination of the cycle-based and the component-based approaches provides a good starting point.

**Table 25:** Comparison of the existing performance and weight evaluation methods

| | **Weight** | **Performance** | **Fast** | **Enable trade-offs** | **Few inputs** |
|---|---|---|---|---|---|
| Cycle-based approach | | ✓✓ | | ✓✓ | ✓ |
| Single parameter approach | ✓✓ | | ✓✓ | | ✓✓ |
| Component-based approach | ✓✓ | | ✓✓ | ✓✓ | ✓ |

However, they both require too many inputs. In addition, the cycle-based approach takes too long to run. These shortcomings can be addressed by leveraging DoEs and RSM.

A DoE is a systematic and efficient process for planning experiments so that the data obtained can be used to establish relationships between the factors affecting a process and the responses of that process. While numerous DoEs exist to generate points in the design space, the Latin Hypercube is selected. This space filling technique enables a rich and highly accurate sampling of the interior of the design space. Once enough data have been

collected, the behavior of the responses is approximated using RSM. The latter generates deterministic relationships between the design variables and the responses. To create these surrogate models, standard least squares regression is used based on the points generated by the DoE. Unlike high-fidelity models, which are long to run, surrogate models allow users to rapidly estimate the response of a process with only small losses of accuracy and with the minimum amount of information necessary. Conducting a sensitivity analysis beforehand also helps reduce the number of design variables necessary to capture the behavior of the system being modeled. This section discusses in detail the implementation of the proposed process followed to create the design framework, as illustrated in Figure 67.

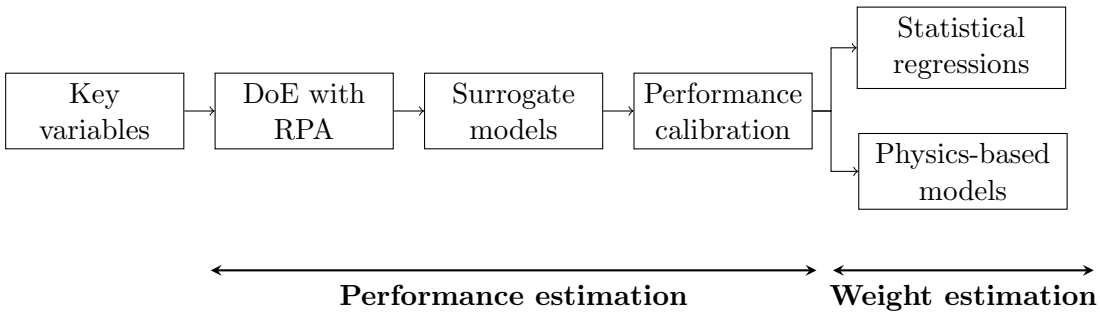



**Figure 67:** Proposed process to develop the design framework

#### *6.3.1.2 Performance Estimation*

The performance of all chemical rocket engines can be described by their vacuum specific impulse $\text{Isp}_v$ and their characteristic velocity $c^*$. The theoretical values of these parameters are first calculated using the cycle-based software RPA [343] as a function of the following key variables: the chamber pressure $p_c$ and the nozzle area expansion ratio $\epsilon$. RPA, an analysis tool suitable for the conceptual and preliminary design of chemical rocket engines, determines chemical equilibrium product concentrations based on the propellants and calculates the thermodynamic properties of the mixture. It outputs the theoretical engine performance using the minimization of the Gibbs free energy. A DoE on RPA is performed to help generate the necessary data. Once this is achieved, surrogate models are created and calibrated to account for the difference between theoretical and actual performance.

**Design of Experiments:** Since the propellant is a categorical variable, one surrogate model per propellant must be developed. The first step consists in selecting all possible propellants to capture the entire design space.

There are two categories of solid propellants: double-base and composite. In double-base propellants, fuel and oxidizer molecules are homogeneously mixed, while in composite propellants they are heterogeneous mixed with binder compounds. Only composite propellants are selected for this research as they have better performance and are widely used in modern engines. Humble [216] mentions three composite propellants that are commonly used: Propellant B (18% of aluminum, 71% of ammonium perchlorate (AP), 11% of hydroxyl-terminated polybutadiene (HTPB)), Propellant C (16% of aluminum, 70% of AP, 14% of polybutadiene acrylonitrile (PBAN)), and Propellant D (22% of magnesium, 66% of AP, 12% of HTPB).

For liquid engines, while both monopropellants and bipropellants exist, only the latter are considered as they give higher performance and are widely used in the industry. They can be classified into three categories: petroleum (mixture of crude oil and complex hydrocarbons), cryogenic (liquefied gases stored at very low temperature), and hypergolic (spontaneously ignitable fuel-oxidizer mixture). Petroleum propellants have lower performance than cryogenic fuels but higher performance than hypergolic fuels. Based on this classification, five liquid propellants are selected. One cryogenic propellant: liquid oxygen ($O_2$ or LOx)/liquid hydrogen ($LH_2$), one petroleum propellant: $O_2$/refined petroleum-1 (RP1) (highly refined kerosene), and three hypergolic propellants: nitrogen tetroxide ($N_2O_4$)/unsymmetrical dimethylhydrazine (UDMH), $N_2O_4$/monomethylhydrazine (MMH), and $N_2O_4$/Aerozine 50 (50% hydrazine and 50% UDMH).

Hybrid engines store the propellant in two different states, usually the fuel is solid and the oxidizer is liquid. Fuels are usually carbon-based polymers in the form of plastic or rubber, typical plexiglass, polyethylene (PE), and HTPB. They can be enhanced with metallic powder such as aluminum to increase their density and thus reduce the overall volume. Many types of liquid oxidizers can be used, including liquid or gaseous oxygen, hydrogen peroxide or nitrous oxide ($N_2O$). The choice of the propellant combination must

be done by weighing quantitative and qualitative considerations: performance, handling, storability, toxicity, etc. Based on a review of the state-of the-art, the selected oxidizers are liquid oxygen, for its performance, and nitrous oxide as it is non-toxic, and easy to handle and store. Paraffin is a fuel that has a high regression rate, allowing high performance. As a consequence, the different combinations of propellants considered are: $O_2$/HTPB, $O_2$/PE, $O_2$/Paraffin, $N_2O$/HTPB, $N_2O$/PE, and $N_2O$/Paraffin.

Based on the 14 selected propellants, Matlab is used to drive RPA and generate the points in the design space. For each propellant, 500 combinations of inputs are generated with $p_c \in [2\text{MPa}\,,\,12\text{MPa}]$ and $\epsilon \in [5\,,\,200]$. RPA is executed for each combination, and the results are stored in a matrix. In addition to $\text{Isp}_v$ and $c^*$, the optimum oxidizer-to-fuel ratio $O/F$ is also tracked for liquid and hybrid propellants. Figure 68 shows the user interface of RPA. Yellow boxes correspond to the required inputs, while the red boxes represent the tracked outputs.

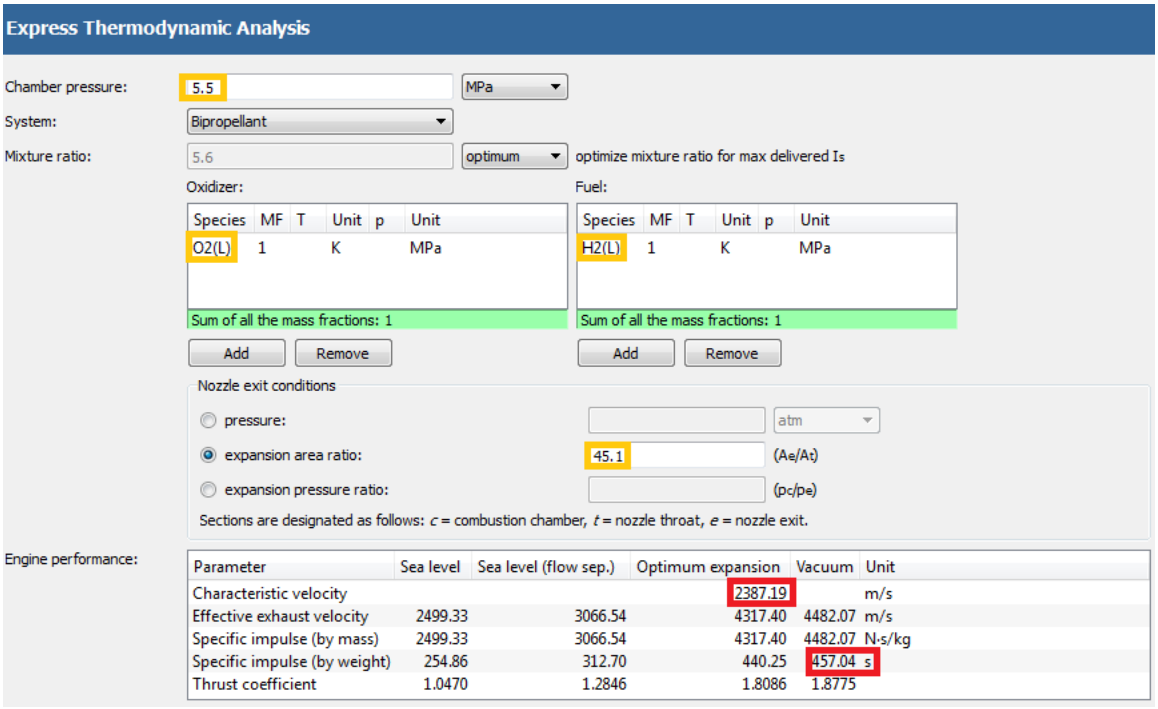



**Figure 68:** User interface of RPA

**Surrogate models:** Based on the aforementioned coverage of the design space, regressions can now be performed. While second-degree polynomial models are commonly used in the literature, they do not provide enough accuracy for the problem at hand. Instead, a logarithmic transformation has been applied to the design variables, hence greatly improving the fits. The general form of the selected surrogate models is presented in Equation 75, where $Y$ is the response, $\alpha_i$ the regression coefficients, and $x_i$ the design variables.

$$Y = \sum_i \alpha_i \ln(x_i) \tag{75}$$

To evaluate the accuracy of the models, four factors are checked: the coefficient of determination $R^2$, the adjusted coefficient of determination $R^2_{adj}$, the Model Fit Error (MFE), and the Model Representation Error (MRE). The mean values for both $R^2$ and $R^2_{adj}$ are 0.955. Both MFEs and MREs show low mean values (between -$10^{-1}$ and $10^{-2}$) and their standard deviations are lower than 0.5. The actual vs. predicted plots have also been checked to ensure a correct behavior of the distribution. A sample of these plots is provided in Appendix B.4.1.2 for each type of propellant.

**Selection of propellants:** The previously developed surrogate models are used to compare the performance of the various propellants. For solid engines, Propellants B, C, and D have similar vacuum specific impulses. The relative difference is smaller than 3% between all propellants. Propellant C was chosen as the representative solid propellant. Tables 26 and 27 compare the $Isp_v$ of all liquid and hybrid propellants. Similar to solid propellants, propellants with a difference in $Isp_v$ smaller than 3% are lumped together. As such, only 3 liquid propellants are kept: $O_2/H_2$, $O_2$/RP1, and $N_2O_4$/MMH. The latter is considered as the representative of all hypergolic propellants. For hybrid propellants, two major groups are identified based on the oxidizer used: $O_2$ or $N_2O$. Because the regression rate of paraffin is higher than that of the other fuels, the following 4 propellants are chosen: $O_2$/HTPB, $O_2$/Paraffin, $N_2O$/HTPB, and $N_2O$/Paraffin. Response surface equations for the selected propellants are provided in Appendix B.4.1.1.

**Table 26:** Difference between the $Isp_v$ of the different liquid propellants in %.

| | $O_2/H_2$ | $O_2$/RP1 | $N_2O_4$/UDMH | $N_2O_4$/MMH |
|---|---|---|---|---|
| $O_2$/RP1 | 20.7 | | | |
| $N_2O_4$/UDMH | 25.1 | 5.6 | | |
| $N_2O_4$/MMH | 24.7 | 5.1 | 0.6 | |
| $N_2O_4$/Aerozine50 | 24.7 | 5.0 | 0.6 | 0.0 |

**Table 27:** Difference between the $Isp_v$ of the different hybrid propellants in %.

| | $O_2$/HTPB | $O_2$/PE | $O_2$/Paraffin | $N_2O$/HTPB | $N_2O$/PE |
|---|---|---|---|---|---|
| $O_2$/PE | 1.8 | | | | |
| $O_2$/Paraffin | 1.8 | 0.0 | | | |
| $N_2O$/HTPB | 13.5 | 15.0 | 15.0 | | |
| $N_2O$/PE | 13.2 | 14.8 | 14.8 | 0.3 | |
| $N_2O$/Paraffin | 13.3 | 14.8 | 14.8 | 0.2 | 0.1 |

**Calibration and validation** Performance parameters provided by RPA represent ideal and theoretical values. In reality, the performance is deteriorated by friction effects, heat transfer, imperfect gases, nonaxial flow, nonuniformity of the fluid, and shifting gas composition. To account for these phenomena, a correction factor $\lambda$ is introduced and calculated by calibrating the performance parameters with existing engines. These calibration factors are determined using the optimization problem described in Equation 76, where $Isp_t$ is the vector of the theoretical values given by the surrogate models and $Isp_a$ is the one composed of actual values found in the literature.

$$\min_{\lambda} \sum_{i} \left| \frac{Isp_a(i) - \lambda Isp_t(i)}{Isp_a(i)} \right| \tag{76}$$

The calibration factor for solid engines is equal to 0.93. For liquid engines, it varies between 0.92 and 0.96. Finally, for hybrid engines, this factor varies between 0.76 and 0.83. These factors are in perfect agreement with various studies [5, 81, 219, 365] comparing theoretical and actual performance. The drop in efficiency for hybrid engines is due to mixing issues

between the liquid oxidizer and the solid fuel. Once the calibration factors have been applied, the average error between the predicted and the actual values is found to be 2.6% over all types of engines. This value is deemed accurate enough at the conceptual and preliminary design stages.

#### *6.3.1.3 Weight/Size Estimation of Solid Engines*

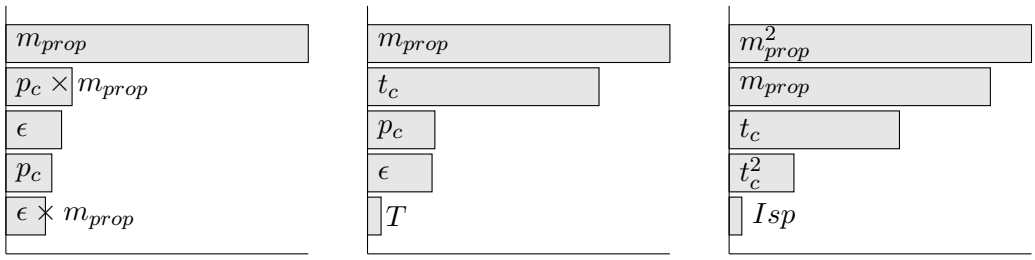
Humble [216] proposes a component-based approach to estimate the weight of solid engines. This approach requires seven input variables and five calibration factors that are unknown at the conceptual design level. The approach has been implemented on three engines whose data are provided by Lara [248]: ORION 50S, GEM 40 VN, and GEM 60. The mean relative error on the dry weight was found to be around 49%, which in unacceptable. The other single parameter approaches [10, 494] also suffer from a lack of accuracy and do not enable trade-off studies. Therefore, new statistical regressions are proposed based on the data presented in Table 28.

Hence, similar to the performance estimation, surrogate models are built to help better estimate the weight of solid engines. A sensitivity analysis is conducted on each parameter to help identify the key design variables. The tornado plots in Figure 69 provide the sensitivity of the weight, length, and diameter with respect to the main key drivers. From this analysis, it appears, as expected, that the mass of propellant $m_{prop}$ is the main driver of the engine weight $m_e$, diameter $D$, and length $L$. The expansion ratio $\epsilon$ mainly impacts the weight and the diameter, which is coherent since a large expansion ratio results in a large and heavy nozzle. The combustion time $t_c$ mainly impacts the engine diameter, which can be explained by the fact that the regression rate has a radial direction.

Based on these design variables, second-order polynomial regressions are performed. The resulting responses are provided in Equation 77. Figure 70 displays the actual vs. predicted plots for each response, along with the key parameters to check for the goodness of fit. These parameters are consistent, and even better, than that of typical statistical aerospace weight equations such as the ones developed by Morrison [185], Glatt [174], Raymer [354], etc.

**Table 28:** Solid engine data [248]

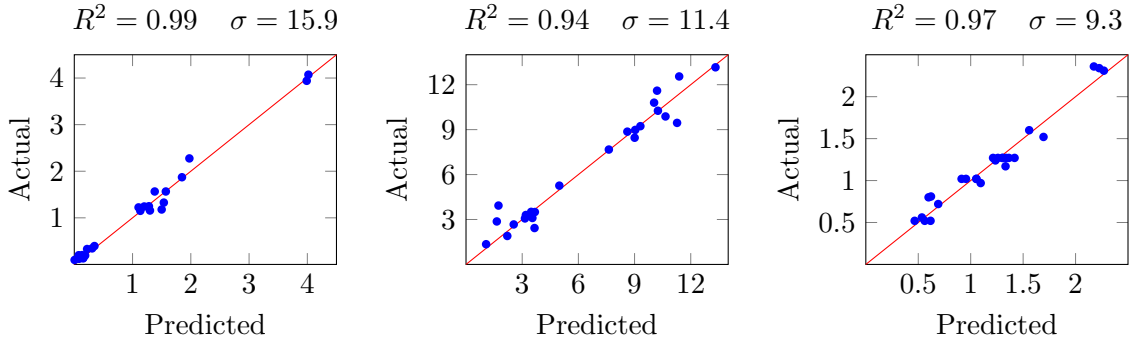
| Engine | $p_c$ (MPa) | $\epsilon$ | $\text{Isp}_\text{v}$ (s) | $m_{prop}$ (kg) | $t_c$ (s) | $T$ (kN) | $m_E$ (kg) | $L$ (m) | $D$ (m) |
|---|---|---|---|---|---|---|---|---|---|
| Castor 120 | 8.59 | 16.3 | 280 | 49,005 | 79.5 | 1,685.9 | 4,072 | 7.67 | 2.36 |
| Orion 50S | 5.61 | 35.3 | 294 | 12,162 | 75.3 | 465.1 | 1,243 | 8.87 | 1.27 |
| Orion 50ST | 5.86 | 26.7 | 286 | 12,157 | 75.0 | 454.4 | 1,249 | 8.46 | 1.27 |
| Orion 50S XL | 7.40 | 34.3 | 294 | 15,023 | 69.1 | 626.3 | 1,150 | 10.26 | 1.27 |
| Orion 50S XLT | 7.47 | 24.8 | 286 | 15,023 | 68.4 | 614.9 | 1,157 | 9.88 | 1.27 |
| Orion 50S XLG | 7.47 | 14.2 | 273 | 15,023 | 68.4 | 588.0 | 1,179 | 9.45 | 1.27 |
| Orion 50 (50T) | 5.58 | 52.1 | 292 | 3,025 | 75.6 | 114.6 | 344 | 2.67 | 1.27 |
| Orion 50 XL | 6.83 | 43.5 | 291 | 3,924 | 69.7 | 160.6 | 395 | 3.10 | 1.27 |
| Orion 38 | 3.94 | 49.3 | 289 | 771 | 67.7 | 32.2 | 121 | 1.35 | 0.97 |
| Orion 32 | 4.55 | 23.0 | 277 | 1,941 | 41.0 | 128.1 | 200 | 3.07 | 0.81 |
| Castor IVA | 4.85 | 8.3 | 268 | 10,109 | 55.2 | 498.3 | 1,565 | 9.23 | 1.02 |
| Castor IVA XL | 4.22 | 15.6 | 282 | 13,112 | 59.4 | 609.9 | 1,871 | 11.60 | 1.02 |
| Castor IVB | 3.16 | 8.0 | 267 | 9,975 | 65.0 | 423.3 | 1,565 | 8.98 | 1.02 |
| Castor 30 | 5.25 | 50.0 | 295 | 12,837 | 143.0 | 258.9 | 1,224 | 3.51 | 2.34 |
| GEM 40 VN | 5.48 | 9.0 | 268 | 11,775 | 64.6 | 478.8 | 1,327 | 10.80 | 1.02 |
| GEM 46 VN | 6.31 | 13.8 | 280 | 16,865 | 76.9 | 601.4 | 2,275 | 12.55 | 1.17 |
| GEM 60 | 5.64 | 11.0 | 274 | 29,697 | 90.8 | 878.7 | 3,940 | 13.16 | 1.52 |
| ASAS 21-85V | 7.58 | 13.9 | 241 | 655 | 24.4 | 62.3 | 96 | 2.43 | 0.52 |
| ASAS 21-120 | 10.20 | 25.0 | 244 | 924 | 22.1 | 99.2 | 130 | 3.50 | 0.52 |
| ASAS 21-120V | 12.40 | 20.0 | 251 | 822 | 17.9 | 110.8 | 192 | 3.30 | 0.52 |
| Oriole | 6.50 | 28.4 | 289 | 976 | 30.0 | 92.5 | 198 | 3.93 | 0.56 |
| ASAS 28-185 | 10.14 | 18.3 | 253 | 2,800 | 29.2 | 231.8 | 331 | 5.26 | 0.72 |
| ASAS 32-58V | 9.58 | 28.0 | 279 | 1,041 | 26.6 | 106.3 | 146 | 1.90 | 0.80 |
| Star 31 | 4.90 | 58.0 | 296 | 1,286 | 45.0 | 82.3 | 108 | 2.87 | 2.29 |
| ORBUS 21 | 4.52 | 63.9 | 296 | 9,707 | 141.6 | 199.8 | 667 | 3.15 | 2.31 |
| Star 48 B | 4.26 | 39.6 | 294 | 2,010 | 85.2 | 68.9 | 131 | 2.03 | 1.24 |




(a) Weight (b) Diameter (c) Length

**Figure 69:** Sensitivity analysis of the weight and size of solid engines

$$\begin{cases} m_e = 1002.0 + 0.10 m_{prop} - 149.55 p_c - 0.03\,(p_c - 6.83)\,(m_{prop} - 7857.27) \\ D = 0.44 + 8.10^{-6} m_{prop} + 0.01 t_c + 0.03 p_c + 0.01\epsilon - 2.10^{-3}\text{Isp} + 4.10^{-7} T \\ L = 25.04 + 5.10^{-4} m_{prop} - 0.08\text{Isp} - 1.10^{-8}\,(m_{prop} - 11333.81)^2 - 8.10^{-4}\,(t_c - 63.83)^2 \end{cases} \tag{77}$$




(a) Weight (tonnes) (b) Length (m) (c) Diameter (m)

**Figure 70:** Goodness of fit of the statistical regressions

Finally, to validate the proposed surrogate models, the mean relative error is computed and compared to existing models. Surrogate models developed for the purpose of this research have a relative error that is much lower than that of existing models. In addition to the significant improvement in accuracy, these models, which use only a handful of design variables, enable engine optimization at the conceptual design phase. Table 29 summarizes these benefits.

**Table 29:** Benefits of the developed model compared to existing methods

| | **Relative error** | **Lower number of design variables** | **Ability to conduct trade-off analysis** |
|---|---|---|---|
| Proposed approach | 15% | ✓ | ✓ |
| Humble [216] | 25% | | ✓ |
| Zandbergen [494] and Akin [10] | 35% | ✓ | |

#### *6.3.1.4 Weight/Size Estimation of Liquid Engines*

A bipropellant liquid rocket engine is composed of a propellant feed system, two tanks to store the fuel and the oxidizer, valves, interconnecting plumbing and components, and a thrust chamber. The latter is made up of the propellant injectors and feed manifold, an igniter, a combustion chamber, an exhaust nozzle, and a structural cooling system. There are two different types of propellant feed systems: tank pressure-fed and pump pressure-fed. Tank pressure-fed systems are usually heavier but less complex than pump pressure-fed systems. Pump pressure-fed systems are chosen for the proposed design environment because they are used for high-thrust applications such as launch and orbit insertion. Based on this physical decomposition, the total engine weight $m_e$ can be calculated using Equation 78, where $m_{tc}$, $m_{tkO}$, $m_{tkF}$, and $m_{st}$ are the masses of the thrust chamber, oxidizer tank, fuel tank, and support structure, respectively.

$$m_e = m_{tc} + m_{tkO} + m_{tkF} + m_{st} \tag{78}$$

Similarly, the length of the engine $L$ is defined in Equation 79, where $l_{tc}$, $l_{tkO}$, and $l_{tkF}$ are the lengths of the thrust chamber, oxidizer tank, and fuel tank, respectively.

$$L = l_{tc} + l_{tkO} + l_{tkF} \tag{79}$$

This section details the models used to compute the weight and size of each of these components.

**Thrust chamber:** To estimate the mass and dimensions of the thrust chamber, historical data of existing liquid engines are used. Correlations between the required thrust $T$ and

the thrust chamber's length $l_{tc}$ and mass $m_{tc}$ are established by Humble et al. [216], and given in Equations 80 and 81.

$$l_{tc} = 3.042 \times 10^{-5} T + 327.7 \tag{80}$$

$$m_{tc} = \frac{T}{g_0(25.2 \log T - 80.7)} \tag{81}$$

**Tanks:** The mass and dimensions of the tanks are estimated using physics-based equations, assuming aluminum cylindrical tanks with semi-spherical ends. First, the masses of the oxidizer $m_{ox}$ and fuel $m_f$ are calculated using Equation 82, where $O/F$ is the optimized oxidizer-to-fuel ratio and $m_{prop}$ the total propellant mass.

$$m_{prop} = m_f + m_{ox} = m_f \left(O/F + 1\right) \tag{82}$$

Each tank is assumed to have a cylindrical section of mass $m_c$ and length $l_c$, and spherical ends of a mass $m_s$ and length $l_s$. Hence, each tank mass can be determined using: $m_{tk} = m_c + m_s$. The mass of the cylindrical section $m_c$ and the spherical ends $m_s$ are calculated using $m_c = A_c t_c \rho_{mat}$ and $m_s = A_s t_s \rho_{mat}$, respectively, where $\rho_{mat}$ is the material density, $A_c$ the surface of the cylindrical section, and $A_s$ the surface of the spherical ends. The wall thickness of the cylindrical part $t_c$ is defined in Equation 83, with $F_{tu}$ the allowable material strength, $p_b$ the burst pressure, and $d_{tk}$ the tank diameter, which is assumed to be fixed and known from airframe constraints.

$$t_c = \frac{0.5 p_b d_{tk}}{F_{tu}} \tag{83}$$

Similarly, the wall thickness of the spherical ends $t_s$ is defined in Equation 84.

$$t_s = \frac{0.25 p_b d_{tk}}{F_{tu}} \tag{84}$$

The burst pressure $p_b$ is calculated using a safety factor $\eta_s$ equal to 2, and a ratio between the maximum expected operating pressure and the tank pressure $\lambda_b$ equal to 1.2. The resulting burst pressure is presented in Equation 85, as provided by Humble et al. [216].

$V_{tk}$ is the volume of each tank determined as a function of the propellant density, assuming a 10% ullage.

$$p_b = \eta_s \lambda_b p_{tk} = \eta_s \lambda_b \left[10^{-0.1068(\log(V_{tk})-0.2588)}\right] \times 10^6 \tag{85}$$

Each tank length can be determined using $l_{tk} = l_c + l_s$. The length of the spherical ends $l_s$ is equal to the diameter of the tank $d_{tk}$ and the length of the cylindrical section $l_c$ is calculated using Equation 86.

$$l_c = \frac{V_{tk} - \frac{4}{3}\pi \left(\frac{d_{tk}}{2}\right)^3}{\pi \left(\frac{d_{tk}}{2}\right)^2} \tag{86}$$

**Support structure:** To take into account the mass of the support structure and attachments $m_{st}$, an empirical relation provided by Rohrschneider [366] is used, as presented in Equation 87, where $T$ is the rocket engine thrust.

$$m_{st} = 0.88.10^{-3} \times (0.225T)^{1.0687} \tag{87}$$

This weight estimation approach is applied to 9 existing liquid engines [38]: Vulcain, RS-68, LE-7A, RL-10A-4-2, RD191, RD0146D, Rocketdyne J2, RD-120, and RD-58M. The results are then compared to actual data for each engine. The average error over the 9 engines is found to be equal to 23% for the engine dry mass, and to 16% for the engine length. These errors are similar to the ones obtained for solid engines and are deemed acceptable at a conceptual design level.

#### *6.3.1.5 Weight/Size Estimation of Hybrid Engines*

A hybrid rocket engine is composed of a tank for the liquid oxidizer, a tank for the pressurization gas, a combustion chamber storing the solid fuel grain protected by an internal insulation, and a nozzle assembly. As such, the total engine mass $m_e$ is calculated using Equation 88, where $m_{tkO}$, $m_{He}$, $m_{tkHe}$, $m_{cc}$, $m_{no}$, and $m_{st}$ are the masses of the oxidizer tank, pressurant gas, pressurant tank, combustion chamber, nozzle, and support structure, respectively.

$$m_e = m_{tkO} + m_{He} + m_{tkHe} + m_{cc} + m_{no} + m_{st} \tag{88}$$

Similarly, the length of the engine $L$ is defined in Equation 89, where $l_{cc}$, $l_{tkO}$, $l_{tkHe}$, and $l_{no}$ are the lengths of the combustion chamber, oxidizer tank, pressurant tank, and nozzle, respectively.

$$L = l_{cc} + l_{tkO} + l_{tkHe} + l_{no} \tag{89}$$

**Oxidizer tank:** The oxidizer tank mass $m_{tkO}$ is calculated using the same method as the one provided for liquid engines. The only difference lies in the calculation of the tank burst pressure $p_b$. For a pressure fed system, the tank burst pressure is defined in Equation 90, where $\delta p_{dynamic} = 0.5\rho_{ox}v^2$ is the oxidizer dynamic pressure (Bernoulli equation with $v$ the flow velocity, typically 10 m/s), $\delta p_{inj} = 0.2p_c$ the injector pressure drop, and $\delta p_{feed} = 50.10^3$ Pa the pressure drop in the feed system (upper range value suggested by Humble [216]).

$$p_b = p_c + \delta p_{dynamic} + \delta p_{inj} + \delta p_{feed} \tag{90}$$

**Pressurant gas:** To be able to assume a constant pressure in the propellant tank, the latter is pressurized with a regulated, inert-gas pressurant. Helium is selected for its low molecular mass. The pressurant gas mass $m_{He}$ is determined using the perfect gas law. Because the combustion time is relatively short and happens in a single burn, an isentropic expansion of the pressurant gas into a constant-pressure propellant tank is assumed. The initial temperature of the gas is assumed to be $T_i = 273$ K and its initial pressure $p_i = 21$ MPa. The final temperature of the gas is then defined in Equation 91 assuming an isentropic change in temperature.

$$T_f = T_i \left(\frac{p_b}{p_i}\right)^{\frac{\gamma-1}{\gamma}} \tag{91}$$

The final volume of pressurant is equal to the volume of the oxidizer tank plus the volume of the pressurant tank. As the volume of the pressurant tank is initially unknown, an iteration is needed.

**Pressurant tank:** To estimate the mass of the pressurant tank $m_{tkHe}$, an empirical approach suggested by Humble [216] is used, involving a tank mass factor $\phi_{tkHe}$ defined in Equation 92.

$$g_0 m_{tk} \phi_{tkHe} = p_i V_{tkHe} \tag{92}$$

The selected value of the tank mass factor is 12,700 m, for a fiber-reinforced composite material.

**Combustion chamber:** The mass of the combustion chamber $m_{cc}$ can be calculated by adding the mass of the case $m_{case}$ and the mass of the injector $m_{inj}$. The injector mass is defined as the mass of a 2.5 cm thick aluminum plate spanning the chamber radius, as presented in Equation 93, where $r_g$ is the chamber radius.

$$m_{inj} = 0.025\rho_{al}\pi r_g^2 \tag{93}$$

The case mass is given in Equation 94, where $l_{cc}$ is the length of the combustion chamber, $t_{cs}$ the thickness of its wall, and $\rho_{mat}$ the material density.

$$m_{case} = 2\pi r_g l_{cc} t_{cs} \rho_{mat} \tag{94}$$

$t_{cs}$ is defined with a safety factor of 1.5 for the chamber burst pressure $p_{bc}$, as presented in Equation 95.

$$t_{cs} = \frac{1.5 p_{bc} r_g}{F_{tu}} \tag{95}$$

The combustion chamber length $l_{cc}$ is calculated by adding the port length $l_p$ to an injector section and an aft mixing section equal to the grain diameter, so the total chamber length is $l_{cc} = l_p + 2r_g$. The radius of the grain $r_g$ is defined in Equation 96, where $w$ is the web thickness and $h$ the height of the port.

$$r_g = h + w + \frac{w}{\sin\theta_p} \tag{96}$$

To determine both $l_p$ and $w$ using Equation 97 and 98, a fuel grain design must be chosen along with the number of combustion ports $N_p$ and their configuration.

$$l_p = \frac{\dot{m}_f}{N_p \rho_f \dot{r}\,(2l + b)} \tag{97}$$

$$w^2 + \frac{(2l + b)\,w}{\pi} = \frac{m_f}{N_p \rho_f l_p \pi} \tag{98}$$

In particular, $l_p$ and $w$ depend on the dimensions of each port defined in the next paragraph and presented in Figure 71: $\theta_p$, $h$, $l$, and $b$.

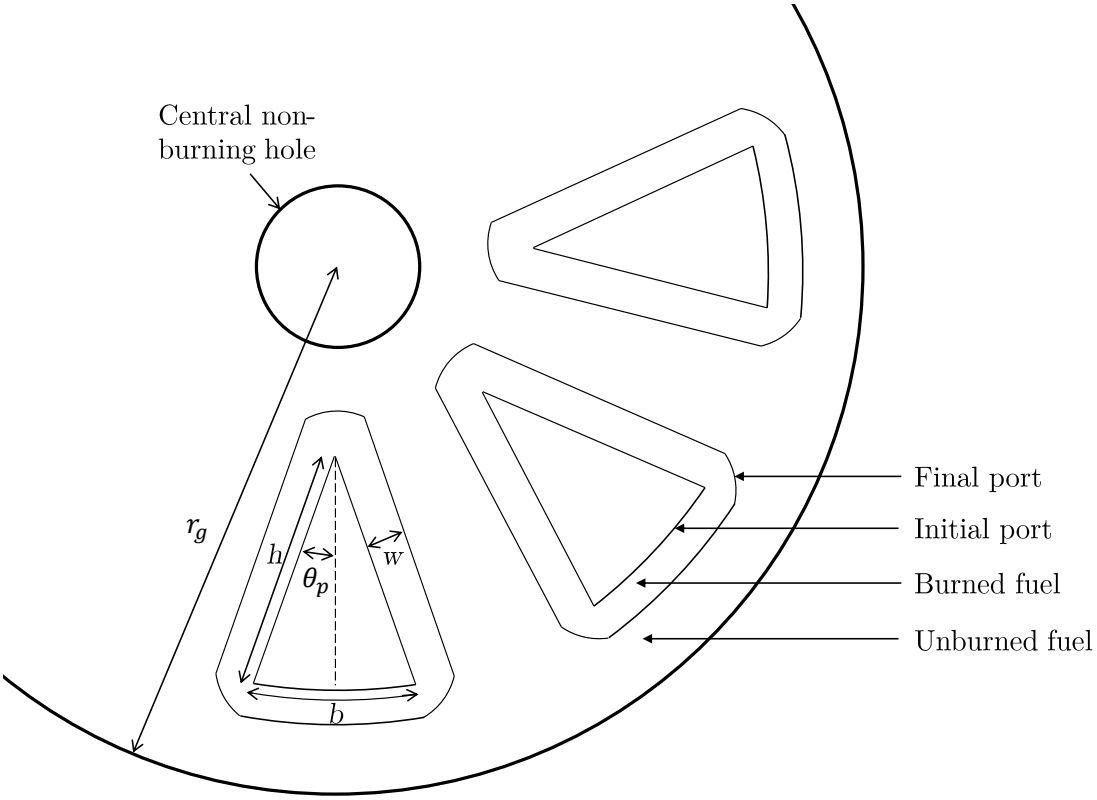



**Figure 71:** Geometric representation of a port

The regression rate $\dot{r}$ is expressed by $\dot{r} = a(G_{oi} + G_{fi})^n$, where $G_{oi}$ represents initial oxidizer mass flux rate and $G_{fi} = G_{oi}/(O/F)$ the initial fuel mass flux rate. The two coefficients $a$ and $n$ are determined experimentally, assuming that the regression rate variation with the grain length is negligible. The values of $a$ and $n$ are presented in Table 30.

**Table 30:** Values of $a$ and $n$ with $\dot{r}$ in mm/s and $G_0$ in kg/m$^2$.s

| | **$O_2$/HTPB [360]** | **$O_2$/Paraffin [230]** | **$N_2O$/HTPB [120]** | **$N_2O$/Paraffin [184]** |
|---|---|---|---|---|
| $a$ | $9.26\times10^{-6}$ | $9.10\times 10^{-5}$ | $1.87\times 10^{-4}$ | $1.32\times 10^{-4}$ |
| $n$ | 0.852 | 0.690 | 0.347 | 0.555 |

The simplest design for the fuel grain is a single cylindrical port. However, this configuration loses efficiency and the $L/D$ ratio becomes quite large as the engine becomes larger. A commonly used wagon-wheel shape with a non-burning center hole and $N_p = 8$ ports is chosen. As discussed by Humble [216], if the number of ports is greater than 7, the

port cross section can be approximated by a triangle. As such, trigonometric considerations can be used to determine the required parameters. The initial port area $A_{pi}$ is defined in Equation 99, where $\dot{m}_{ox}$ is the oxidizer flow rate.

$$A_{pi} = \frac{\dot{m}_{ox}}{N_p G_{oi}} \tag{99}$$

The half angle of each port is $\theta_p = \pi/N_p$, the triangle height is $h = (A_{pi}/\tan\theta_p)^{0.5}$, the length of the triangle base is $b = 2h\tan\theta_p$, and the length of the side is $l = h/\cos\theta_p$.

**Nozzle:** Assuming a bell nozzle with a half angle $\theta_n = 15°$, the length of the nozzle is defined in Equation 100, where $D_t$ is the throat diameter and $D_e$ the exit diameter.

$$l_{no} = 0.8\frac{D_e - D_t}{2\tan(\theta_n)} \tag{100}$$

The throat diameter $D_t$ can be calculated using the throat area $A_t$ from the characteristic velocity equation, as presented in Equation 101, where $\dot{m}$ is the propellant mass flow rate.

$$A_t = \frac{\dot{m}c^*}{p_c} \tag{101}$$

Then, the exit diameter $D_e$ is related to the throat diameter via the expansion ratio $\epsilon$. To estimate the mass of the nozzle, an empirical relationship assuming a phenolic nozzle provided by Humble [216] is used, as presented in Equation 102.

$$m_{no} = 125\left(\frac{m_{prop}}{5400}\right)^{2/3}\left(\frac{\epsilon}{10}\right)^{1/4} \tag{102}$$

**Support structure:** According to Humble [216], the mass of the support structure and ancillary parts $m_{st}$ represent 10% of the total inert mass of the engine.

This weight estimation method has been applied with data from the hybrid engine of the suborbital vehicle SpaceShipOne. The total mass of the unfuelled engine was found to be 50% of the empty mass of the vehicle and the length 63% of the total length of the vehicle. This is in perfect agreement with multiple studies [157, 378, 382].

#### *6.3.1.6 Summary*

As described in this section, the proposed design environment provides the capabilities to rapidly estimate the performance, weight, and size of all types of chemical rocket engines. This corresponds to Experiment 3.1. Section 6.3.1.2 shows that, on average, the mean relative error for performance evaluation is 2.6%, which is highly accurate for a conceptual design level. Then, Sections 6.3.1.3 to 6.3.1.5 demonstrate that the average relative error on the weight/size estimation is around 18%, which is better than other conceptual design level weight estimation tools [10, 216, 494]. In addition, the number of design variables is equal to seven, which is acceptable for further trade-off analyses and design space exploration. Finally, the computation time of the proposed methodology is around several milliseconds against several seconds using RPA combined with an existing weight estimation approach. This validates the four criteria of Experiment 3.1 and consequently Hypothesis 3.1.

**VALIDATION HYPOTHESIS 3.1: IF performance parameters found by creating surrogate models of the cycle-based software Rocket Propulsion Analysis (RPA) are inputted into a physics-based weight prediction model THEN performance and weight of liquid, solid, and hybrid rocket engines can be rapidly predicted at a conceptual design level.**

### 6.3.2 Jet Engines

This section aims at modeling both the performance and the weight of jet engines at a conceptual design level. Raymer provides a series of statistical equations, listed in Appendix B.4.2, that estimate the empty weight $W$, length $L$, diameter $D$, and TSFC of all jet engines as a function of four main variables [354]. Table 31 describes the input variables along with their ranges.

In addition to the bypass ratio and the required thrust, one of the key performance drivers in jet engines is the allowable turbine inlet temperature. This parameter has also an important impact on the engine cost. Indeed, a high turbine inlet temperature enables a

**Table 31:** Input variables for jet engine modeling

| Variables | Descriptions | Ranges |
|---|---|---|
| T | Take-off thrust (kN) | [0 , 100] |
| BPR | Bypass ratio | [0 , 1] with afterburners<br>[0 , 6] without afterburners |
| M | Maximum Mach number | [0 , 2.5] with afterburners<br>[0 , 1] without afterburners |
| Afterburners | Presence of afterburners | Yes - No |

better combustion and consequently increases the efficiency of the engine. However, it also increases the cost due to the use of state-of-the-art technologies. In order to account for this trade-off, relationships between TSFC and $T_4$ must be modeled and integrated in the previous performance evaluation model. Boggia and Rud [42] have assessed the impact of $T_4$ on the engine fuel consumption as a function of the bypass ratio. Figure 72 shows the results of this analysis.

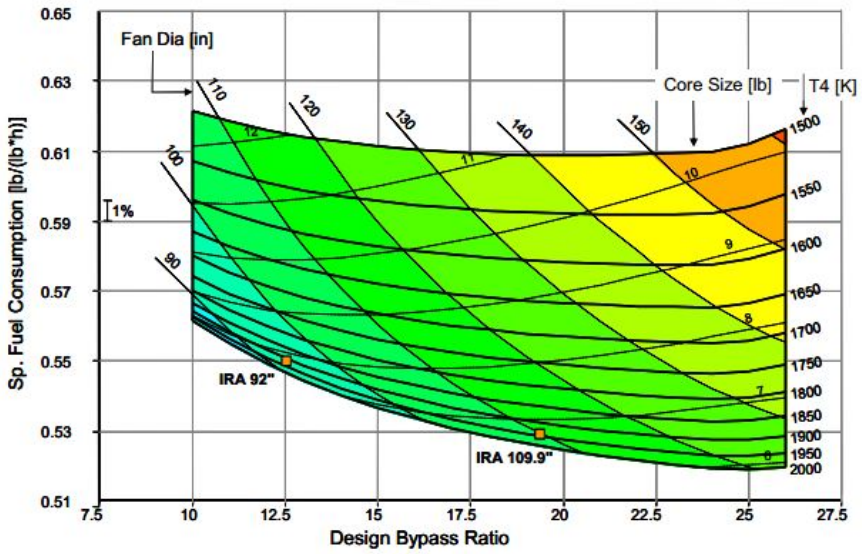



**Figure 72:** Impact of the turbine inlet temperature on the TSFC [42]

To integrate this impact into the design framework, a mathematical model is developed using polynomial approximations with a baseline $T_4$ fixed at 1,500 K. Then, the impact of a relative change in $T_4$ on the TSFC is captured through multiple points in the provided

simulation. Based on these points, a second order RSE is generated. It is presented in Equation 103, where $\Delta T_4$ is the relative change in the turbine inlet temperature compared to the baseline and $\Delta TSFC$ its relative reduction on the TSFC in percentage.

$$\Delta TSFC = -0.51 + 24.35\Delta T_4 + 0.30BPR + 0.83\,(\Delta T_4 - 0.23)\,(BPR - 17.50) \tag{103}$$

As shown in Equation 104, the gradient in $\Delta TSFC$ is positive along all directions for positive changes in $T_4$ and positive BPR. Hence, an increase in $T_4$ and/or in $BPR$ would improve the fuel efficiency of the engine.

$$\nabla \Delta TSFC(T_4, BPR) = \begin{pmatrix} 24.35 + 0.83(BPR - 17.50) \\ 0.30 + 0.83(\Delta T_4 - 0.23) \end{pmatrix} \tag{104}$$

In this research, $T_4$ is added to the list of design variables for the jet engine and varies between 1,500 K and 2,000 K.

This section modeled all possible types of propulsion systems that can be used for suborbital flights. Combined with the previous section, two of the external forces acting on the vehicle can be modeled: weight and thrust. The last two forces (lift and drag) are discussed in the following section.

## *6.4 Aerodynamic Modeling*

The aerodynamic modeling aims at developing a prediction model for the aerodynamic lift and drag coefficients in all flight regimes and for all configurations. As discussed in Section 3.3.4, the model is based on the one developed by Roskam [369]. While both the subsonic and the supersonic regimes are well-known and can be well predicted, the transonic regime is harder to model. Indeed, complex flow phenomena make the drag coefficient difficult to estimate. Raymer suggests a conceptual level approach based on experimental measurements for the flow between Mach 0.6 and Mach 1.2. This section first describes the approach for estimating the lift coefficient. Then, the different models for each drag component and each flight regime are developed. Finally, multiple simplifications are discussed in order to speed up the process.

### 6.4.1 Lift Coefficient Model

The vehicle lift coefficient $C_L$ can be decomposed into two different terms, as defined in Equation 105. In this equation, $\alpha$ represents the angle of attack and $C_{L_\alpha}$ the lift-curve slope.

$$C_L = C_{L_\alpha}\alpha \tag{105}$$

In this research, it is assumed that only the wing can produce lift. While the angle of attack depends on the flight conditions, the lift-curve slope can be modeled as a function of the Mach number and the wing configuration. In the subsonic regime, Raymer suggests the semi-empirical formula presented in Equation 106, where $AR$ is the aspect ratio, $\beta = \sqrt{1-M^2}$ the compressibility factor, $S_{exposed}$ the surface of the wing exposed to the air, $\Lambda$ the sweep angle, $\eta$ the airfoil efficiency factor, $F$ the fuselage lift factor [354], and $S$ the reference area.

$$C_{L_\alpha} = \frac{2\pi AR}{2 + \sqrt{4 + \frac{AR^2\beta^2}{\eta^2}\left(1 + \frac{\tan^2\Lambda}{\beta^2}\right)}}\frac{S_{exposed}}{S}F \tag{106}$$

This equation is assumed to be valid up to the drag divergence Mach number $M_{DD}$. The models and values selected for the different parameters are described in Appendix B.3.1.

For the supersonic speed, usually beyond Mach 1.2, the theoretical lift-curve slope is defined in Equation 107, corrected by an efficiency factor $\eta_s$ to match actual data [354].

$$C_{L_\alpha} = \frac{4}{\sqrt{M^2-1}} \tag{107}$$

The efficiency factor $\eta_s$ is defined in Equation 108, where $C_{L_\alpha}(M_{DD})$ is the value of the lift-curve slope at the drag divergence Mach number.

$$\eta_s = 0.141 C_{L_\alpha}(M_{DD}) \tag{108}$$

According to Raymer, there are no good initial-estimation methods for the transonic regime and it is suggested to smoothly link both the subsonic and the transonic regimes. To do so, a parametric interpolation is developed based on experimental data [354] and is presented in Appendix B.3.1.

Figure 73 shows the results of the previous model by plotting the lift-curve slope as a function of the Mach number and the sweep angle. The plot confirms that, the higher the sweep angle, the lower the lift coefficient at a given angle of attack.

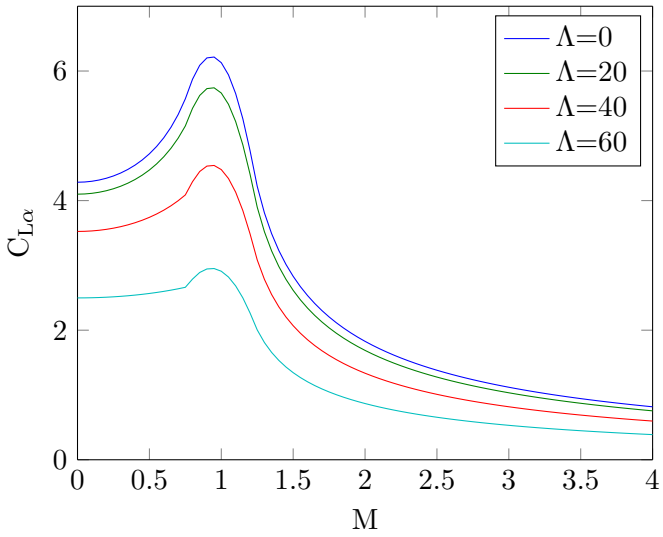



**Figure 73:** $C_{L_\alpha}$ with respect to the Mach number and the sweep angle

### 6.4.2 Maximum Lift Coefficient Model

The maximum lift coefficient $C_{L,max}$ is a crucial parameter for winged bodies, especially during take-off and landing. According to Raymer [354], it can be determined using Equation 109, where $C_{l,max}$ is the maximum lift coefficient of the airfoil and $\Lambda$ the sweep angle.

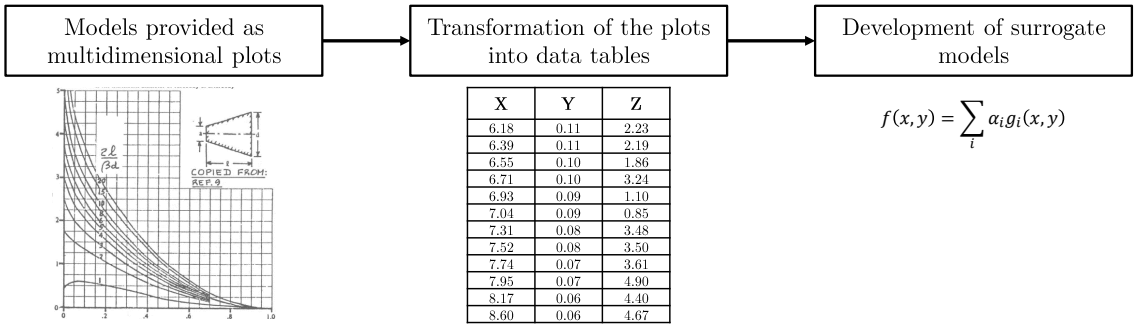
$$C_{L,max} = 0.9 C_{l,max} \cos \Lambda \tag{109}$$

Assuming single-slotted flaps, Hoerner provides the value of $C_{l,max}$ for critical phases: $C_{l,max} = 2.8$ during take-off and $C_{l,max} = 3$ during landing [206].

### 6.4.3 Subsonic and Supersonic Drag Coefficient Models

Roskam decomposes the drag coefficient into 11 components: wing drag coefficient ($C_{D_{wing}}$), fuselage drag coefficient ($C_{D_{fus}}$), empennage drag coefficient ($C_{D_{emp}}$), nacelle/pylon drag coefficient ($C_{D_{np}}$), flap drag coefficient ($C_{D_{flap}}$), landing gear drag coefficient ($C_{D_{gear}}$), canopy/windshield drag coefficient ($C_{D_{cw}}$), store drag coefficient ($C_{D_{store}}$), trim

drag coefficient ($C_{D_{trim}}$), interference drag coefficient ($C_{D_{int}}$), and miscellaneous drag coefficient ($C_{D_{misc}}$). The latter includes the drag caused by speed brakes, struts, inlets, antennas, gaps, and surface roughness. In addition to this first decomposition, the model for the drag coefficient is specific to each flight regime: subsonic and supersonic. For each combination of component and flight regime, the multidimensional plots provided by Roskam are used to extract data points that cover the entire design space. Based on these data points, surrogate models are developed in order to create parametric relations to estimate the various parameters needed to determine the vehicle drag coefficient. Figure 74 provides an overview of this process, whose the implementation is further discussed in Appendix B.3.2.




| X | Y | Z |
|---|---|---|
| 6.18 | 0.11 | 2.23 |
| 6.39 | 0.11 | 2.19 |
| 6.55 | 0.10 | 1.86 |
| 6.71 | 0.10 | 3.24 |
| 6.93 | 0.09 | 1.10 |
| 7.04 | 0.09 | 0.85 |
| 7.31 | 0.08 | 3.48 |
| 7.52 | 0.08 | 3.50 |
| 7.74 | 0.07 | 3.61 |
| 7.95 | 0.07 | 4.90 |
| 8.17 | 0.06 | 4.40 |
| 8.60 | 0.06 | 4.67 |

**Figure 74:** Drag coefficient estimation process

### 6.4.4 General Transonic Drag Coefficient

Due to the lack of accurate analytical or numerical approaches for the transonic regime, Raymer suggests an empirical approach [354] to predict the drag coefficient between Mach 0.6 and Mach 1.2. This approach is based on several rules presented below:

- The drag coefficient at Mach 1.2 is equal to the drag coefficient at Mach 1.05.
- The drag coefficient at Mach 1 is half the drag coefficient at Mach 1.05.
- The drag coefficient has a smooth behavior over the entire transonic regime.
- The drag coefficient has a linear behavior between Mach 1 and Mach 1.05.

Following these rules, a parametric approach is developed using the aforementioned models to compute the vehicle's drag coefficient at Mach 0.6 and 1.2. Then, the transonic range

is decomposed into three different ranges where different models are applied: from Mach 0.6 to Mach 1, from Mach 1 to Mach 1.05, and from Mach 1.05 to mach 1.2. The detailed development of the three models are discussed in Appendix B.3.3.

### 6.4.5 Simplifications

The proposed aerodynamic model is intended to be used to optimize the trajectory. While this model is highly accurate, it needs to be run for each combination of altitude and Mach number during the trajectory optimization. It appears to be time consuming when used for complex multi-objective optimization purposes. As a consequence, a series of approximations are made in order to speed up the overall process. The general idea is to avoid the entire model to be run at each iteration of the trajectory optimization. To do so, several key parameters are computed once and then used to create simple and accurate surrogate models as a function of altitude and Mach number at a vehicle level. The dependence of the lift drag coefficient of a typical vehicle with respect to both altitude and Mach number is displayed in Figure 75. This drag polar has been calculated with the available data from the SpaceShipTwo.

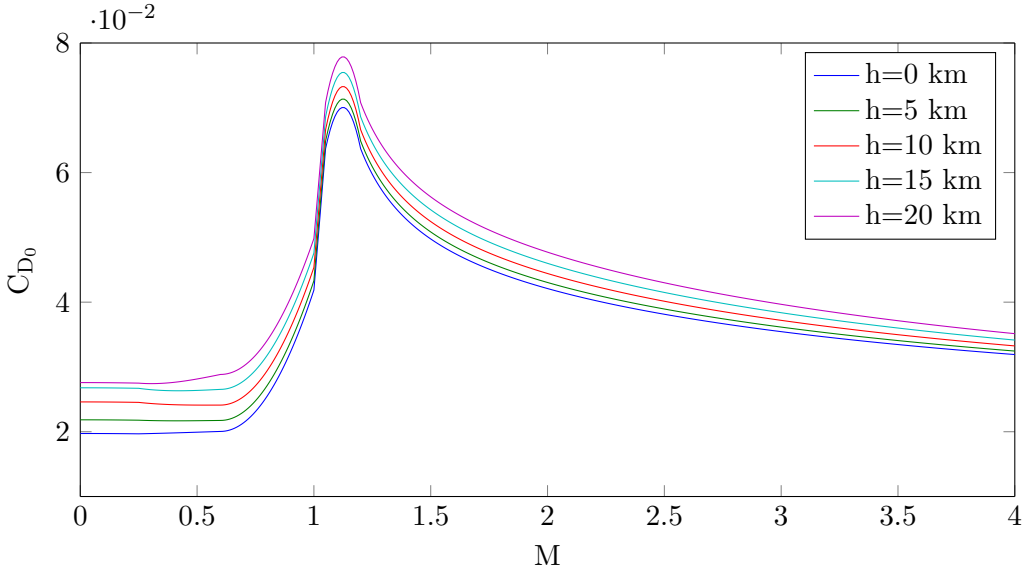



**Figure 75:** $C_{D_0}$ with respect to the Mach number and for different altitudes

Using Figure 75, several observations can be made. First, three distinct regions with specific behaviors are identified in terms of Mach number dependence. Indeed, from Mach

0 to Mach 0.6, the drag coefficient is almost constant and can be modeled with a single coefficient. From Mach 0.6 to Mach 1.2, the behavior is more complex but its behavior has been well described by Raymer and is already based on surrogate models. Finally, the supersonic region follows a very smooth curve that can be modeled by a rational fraction. Multiple interpolations with Matlab show that Equation 110 can be used with a $R^2$ of around 0.995, where $a$, $b$, $c$, and $d$ are four constants to be parametrically determined.

$$C_D(M) = \frac{aM + b}{M^2 + cM + d} \tag{110}$$

In addition, for all Mach numbers, the dependence of altitude is close to a linear model. Based on these observations, seven coefficients have to be defined by the aerodynamic module in order to compute the lift drag coefficient for any combination of altitude and Mach number for a given vehicle configuration. The five-step process proposed below is presented below.

1. Calibrate the aerodynamic model for a given configuration.

2. Compute the drag coefficient of the vehicle for five different Mach numbers: 0.6, 1.2, 3, 5, and 7 and two different altitudes: 0 and 20 km.

3. Determine the coefficients of the surrogate models for supersonic speeds as a function of the drag coefficients at Mach 1.2, 3, 5, and 7.

4. Determine the regression coefficients of the linear interpolation of the drag coefficient for a given Mach number as a function of altitude.

5. Pass the coefficients to the trajectory calculation program.

This section discussed the development of the aerodynamic module, which is able to rapidly evaluate all aerodynamic coefficients. This module is then combined with the ones developed in Sections 6.2 to 6.3 to provide all required information to optimize the vehicle trajectory, as discussed in the next section.

## *6.5 Trajectory Modeling*

The objective of the trajectory module is to calculate the amount of fuel required for a given vehicle to meet the requirements. Because the trajectory module is to be integrated into a large and complex optimization algorithm, it needs to be fast to run. As discussed in Section 3.3.5, a physics-based model is developed in order to optimize and simulate the vehicle's trajectory. Indeed, this approach represents the best trade-off between execution time and accuracy. First, the approach used to model the traditional aircraft mission phases is provided: take-off, landing, and climb. Then, the principles governing the ESA approach are described along with the assumptions specific to suborbital vehicles. The implementation and the validation of this approach into Matlab are also detailed. Finally, the module is applied to optimize and simulate the trajectory of existing vehicles.

### 6.5.1 Take-Off and Landing

For horizontal take-off and/or horizontal landing vehicles, there is a need to ensure that the required runway length meets the spaceport's constraints. As a consequence, it is necessary to assess the take-off and landing performance of the vehicle.

#### *6.5.1.1 Take-Off*

The take-off distance can be decomposed into four portions: ground roll, rotation, transition, and initial climb to clear the obstacle. The detailed calculation of each of these distances requires technical data that are not available at a conceptual design level. Instead, the "magic line" approach, presented by Shevell [395], that consists in a simplified formula to estimate the take-off field length, is used. Using a 15% margin to account for safety requirements, the take-off field length $L_{TO}$ is defined in Equation 111 [43], where $C_{L,max}$ is the maximum lift coefficient, $W_{TO}$ the take-off gross weight, and $T_{0.7}$ the overall thrust delivered at 70% of the take-off speed.

$$L_{TO} = \frac{0.17 W_{TO}^2}{\left(\frac{\rho}{\rho_0}\right)^{0.8} S C_{L,max} T_{0.7}} + 27 \tag{111}$$

#### *6.5.1.2 Landing*

Similar to the take-off distance, the landing distance can be decomposed into three phases: approach, free roll, and braking. A simplified approach presented by Boiffier [43] and similar to the "magic line" is used. Assuming that the aircraft is equipped with an Antilock Braking System (ABS) system and that only the braking force is considered along the horizontal axis, the landing distance is given in Equation 112, which includes a 15% safety margin. In this equation, $V_s$ is the stall speed and $\mu_r = 0.5$ the friction coefficient.

$$L_{LA} = \frac{1.69 V_s^2}{2 \mu_r g_0} \tag{112}$$

### 6.5.2 Climb with Jet Engines

The climb phase with jet engines is assumed to be performed at the maximum climb rate. This assumption, which is commonly used for commercial aircraft, results in a climb that reduces the time spent in this phase. Based on this assumption, the maximum rate of climb $ROC_{max}$ is defined in Equation 113, where $Z$ is defined in Equation 114 [13].

$$ROC_{max} = \sqrt{\frac{WZ}{3S\rho C_{D_0}}} \left(\frac{T}{W}\right)^{\frac{3}{2}} \left(1 - \frac{Z}{6} - \frac{3}{2Z\left(\frac{L}{D}\right)_{max}^2 \left(\frac{T}{W}\right)^2}\right) \tag{113}$$

$$Z = 1 + \sqrt{1 + \frac{3}{\left(\frac{L}{D}\right)_{max}^2 \left(\frac{T}{W}\right)^2}} \tag{114}$$

The corresponding speed $V_{ROC_{max}}$ is defined in Equation 115.

$$V_{ROC_{max}} = \sqrt{\frac{T}{3S\rho C_{D_0}} \left(1 + \sqrt{1 + \frac{3}{\left(\frac{L}{D}\right)_{max}^2 \left(\frac{T}{W}\right)^2}}\right)} \tag{115}$$

As shown by the previous equations, the optimal flight conditions depend on altitude and weight. Hence, an iterative process is developed to calculate the corresponding fuel consumption from the ground to the transition altitude. For that purpose, at each iteration, Equation 116 is used to compute the fuel consumption during a single iteration $\Delta W$, where $n_{Jet}$ is the number of jet engines, $T$ the thrust delivered by a single jet engine at the given flight conditions (altitude and speed), $\Delta t$ the duration of the iteration, and $TSFC$ the jet

engine thrust specific fuel consumption.

$$\Delta W = TSFC \times \Delta t \times T \times n_{Jet} \tag{116}$$

### 6.5.3 Ascent with Rocket Engine

In order to calculate the ascent trajectory with the main propulsion type, the ESA is used. This approach is based on the definition of the total energy per unit mass $E_s$ as a state variable. Its objective is to compute the fuel required to go from one energy state to another one.

#### *6.5.3.1 Problem Formulation*

The specific total energy $E_s$ is defined in Equation 117, where $V$ is the velocity and $h$ the altitude. It sums the contributions of both the potential energy and the kinetic energy.

$$E_s = g_0 h + \frac{V^2}{2} \tag{117}$$

Based on this definition and the general equations of motion presented in Section 3.3.5, the time rate of change of the total energy per unit mass $\dot{E}_s$ can be calculated using Equation 118, where $T$ is the thrust, $D$ the drag, $m$ the mass, and $\epsilon$ the thrust angle.

$$\dot{E}_s = V \frac{T \cos \epsilon - D}{m} \tag{118}$$

The fuel consumption is modeled by the time rate of change in the vehicle's mass $\dot{m}$, as defined in Equation 119, where $Isp$ is the specific impulse of the propulsion system.

$$\dot{m} = -\frac{T}{g_0 Isp} \tag{119}$$

Combining Equations 118 and 119, the relative change in the vehicle mass is defined in Equation 120.

$$\frac{dm}{m} = -\frac{T}{g_0 Isp \left(T \cos \epsilon - D\right) V} dE \tag{120}$$

Based on this equation, the optimum ascent trajectory between two energy levels in terms of fuel consumption can be found by minimizing the objective function $F(M, h)$ defined in Equation 121.

$$F(M, h) = \frac{g_0 Isp \left(D - T \cos \epsilon\right) V}{T} \tag{121}$$

Once the optimum point has been found for each energy level, the corresponding fuel consumption can be determined using Equation 120.

In addition, multiple design constraints must be met in order to output a feasible solution. These constraints are described in the next section.

#### *6.5.3.2 Design Constraints*

The theoretical optimum fuel-to-climb trajectory can be determined using the aforementioned problem formulation. However, some constraints must be set on the flight trajectory to account for the structural and thermal limitations of the vehicle, as well as the physical limitations of the passengers. According to Clervoy et al. [214, 252], three major constraints must be considered for suborbital vehicles: maximum dynamic pressure, maximum load factor, and maximum temperature.

**Maximum dynamic pressure:** For high-speed flights, the kinetic energy of the fluid particles intercepting the vehicle is extremely high. These particles create important efforts that are applied to the structure and are proportional to the dynamic pressure. As such, the structural design of the vehicle is mainly driven by the dynamic pressure. The latter must thus be limited in order to ensure the structural integrity of the vehicle. For suborbital vehicles' technologies, the typical maximum admissible dynamic pressure is 50 kPa [214, 252]. This limitation is described in Equation 122, where $q_{max}$ is the maximum admissible dynamic pressure and $\rho$ the density.

$$q = \frac{1}{2}\rho V^2 \leq q_{max} \tag{122}$$

**Maximum load factor:** Since suborbital vehicles aim at carrying passengers, there is a need to limit the maximum load factor. Indeed, the load factor directly affects human bodies, as discussed in Section 1.3.1.1. Two different types of load factor exist: longitudinal and vertical. However, for suborbital flight, only the longitudinal load factor must be considered. Indeed, both the ascent and descent phases occur at high flight path angle: close to 90°. As a consequence, all accelerations and decelerations are in a direction parallel to the vehicle longitudinal axis. The mathematical formulation of this constraint is presented

in Equation 123, where $T$ is the thrust, $\epsilon$ the thrust angle, $D$ the drag, $m$ the vehicle's mass, $\gamma$ the flight path angle, and $n_{max}$ the maximum admissible load factor.

$$n = \frac{T\cos\epsilon - D}{mg} - \sin\gamma \leq n_{max} \tag{123}$$

The value of this maximum admissible load factor has been modeled in Section 4.2.

**Maximum temperature:** Thermal constraints that must be taken into account during a suborbital flight are directly related to the Mach number and the air temperature. While most of the thermal energy is dissipated thanks to the shock generated in front of the vehicle at supersonic speeds, there is still a need for ensuring that the wall temperature does not exceed its maximum admissible temperature. When coupled with a TPS, typical materials used for suborbital vehicles can handle temperatures up to $T_{max} = 1,000$ K [252]. The constraint for the maximum temperature is described in Equation 124, where $T_\infty$ is the temperature of the incoming air, $M$ the Mach number, and $T_{max}$ the maximum admissible temperature.

$$T = T_\infty + 0.2T_\infty M^2 \leq T_{max} \tag{124}$$

#### *6.5.3.3 Assumptions*

In order to reduce the computational time, a major assumption suggested by Bryson et al. [61] is to neglect the angle of attack in the optimization process. Indeed, for rocket-powered phases, this angle is usually very close to 0 in order to limit the drag, since the trajectory is exclusively controlled by the thrust direction.

#### *6.5.3.4 Development and Validation*

Using the aforementioned approach, the trajectory is calculated by solving a series of optimization problems. The trajectory is decomposed into $n$ steps in the energy space starting at $E_s(M_i, h_i)$ and ending at $E_s(M_f, h_f)$, where $h_i$ and $M_i$ are the initial altitude and Mach number, respectively, and $h_f$ and $M_f$ are the final altitude and Mach number, respectively. For each step $k$, the optimization problem to be solved is presented below,

where Equation 125a is the objective function and Equation 125b the constraints.

$$\min_{M,h} \frac{g_0 Isp\,(D - T\cos\epsilon)\,V}{T} \tag{125a}$$

$$\begin{cases} T_\infty + 0.2T_\infty M^2 \leq T_{max} \\ \frac{T\cos\epsilon - D}{mg} - \sin\gamma \leq n_{max} \\ \frac{1}{2}\rho V^2 \leq q_{max} \\ \frac{V^2}{2} g_0 h = E_s(k) \end{cases} \tag{125b}$$

The architecture of the trajectory module is presented in Figure 76.

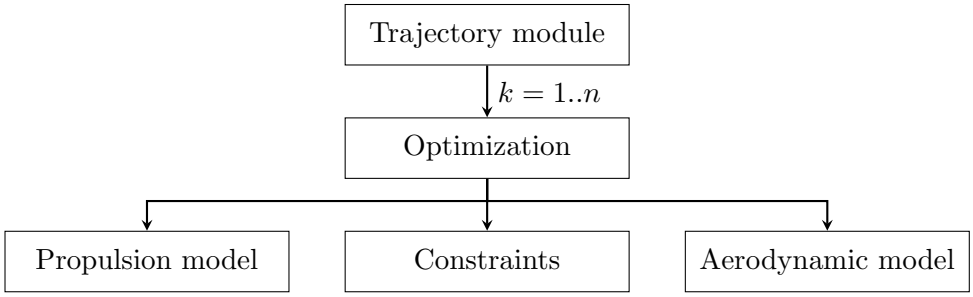



**Figure 76:** Architecture of the trajectory module

The main function drives the optimization loop across the successive steps in the energy space. For each step, the trajectory module calls an optimization function that minimizes the previously described objective function, while also ensuring that the constraints are met. Since the vehicle's characteristics depend on the flight conditions, a simplified version of both the aerodynamic and propulsion modules must be called. The details about the implementation are presented in Appendix D.2.5.

In order to validate the implemented model, the module is used to compute the performance of a supersonic aircraft that has already been modeled by Bryson et al. [61]. The obtained trajectory, presented in Figure 77, is similar to the one calculated in the original research, shown in Figure 78.

The discontinuities at Mach 0.8 and 1.2 are due to approximations in the piecewise aerodynamic model of the vehicle. The behavior and the values of the key points in the trajectory are very similar to the ones provided by Bryson [61].

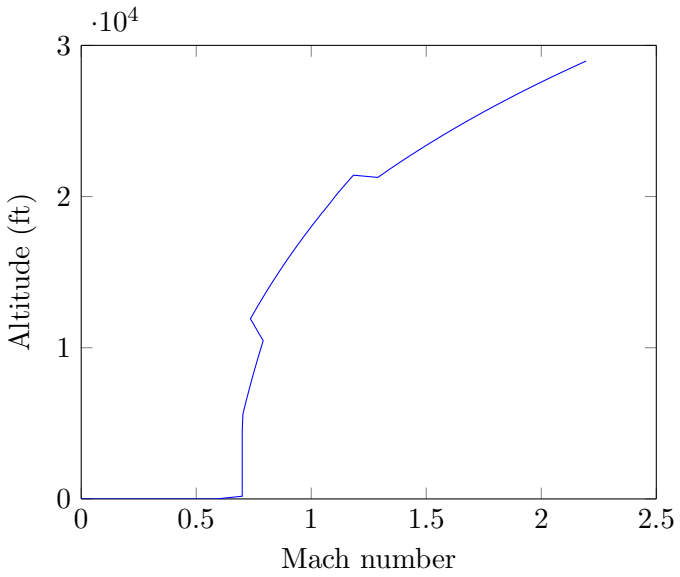



**Figure 77:** Calculated optimum fuel-to-climb trajectory of the test aircraft

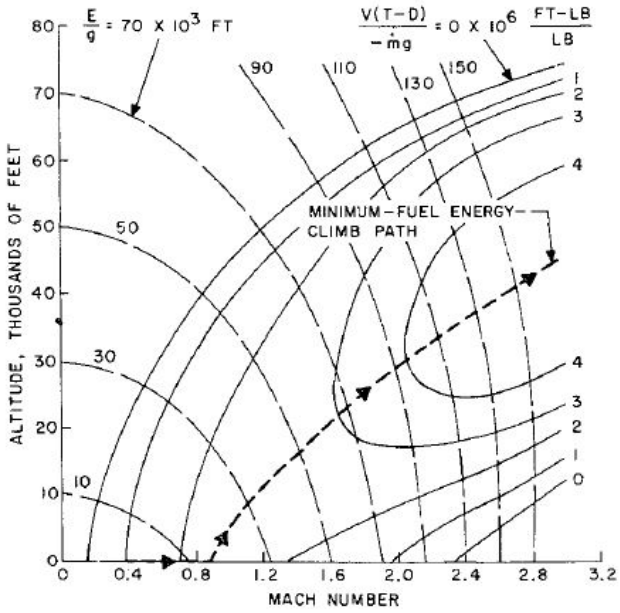



**Figure 78:** Theoretical optimum fuel-to-climb trajectory of the test aircraft [61]

### 6.5.4 Application to Suborbital Vehicles

The aforementioned approach has been implemented and tested for the SpaceShipTwo, which is launched from a carrier aircraft at an altitude of about 14 km. The optimum fuel-to-climb trajectory is presented in Figure 79.

The behavior of the trajectory that minimizes the fuel consumption is very similar to

the ones of supersonic fighter aircraft [13]. Indeed, after performing an initial accelerated climb, the vehicle descends around Mach 1 in order to rapidly gain speed. This reduces the time spent in the transonic regime, where the drag increase can be very high.

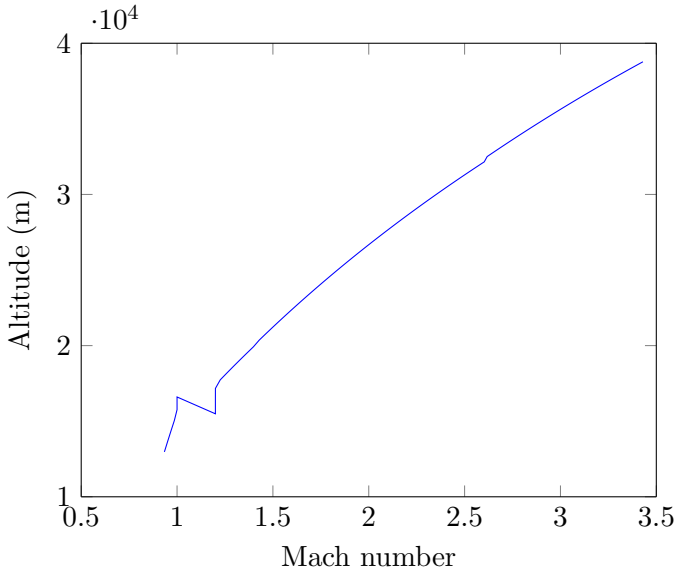



**Figure 79:** Optimum fuel-to-climb trajectory of the SpaceShipTwo

Sections 6.2 to 6.5 brought together the required pieces to assess the flying performance of the vehicle. However, the design and engineering of a new product also require economic considerations. Hence, the following section discusses the different approaches used to estimate the various life-cycle cost components.

## *6.6 Life-Cycle Cost Estimation*

This section aims at developing an environment for calculating the life-cycle costs of all possible suborbital vehicles. As discussed in Section 3.3.6, the major sources of Cost Estimating Relationships (CERs) are Suborb-TransCost [176], studies from Morrison et al. [182, 185], and from Nieroski et al. [320]. However, a new approach has to be defined to estimate the life-cycle costs of hybrid rocket engines. This section starts with general economic considerations that must be taken into account when developing a cost estimation model. Then, the various cost categories considered in this study are described along with the relationships between each other. Finally, the estimation approaches used for each of

these categories are described and validated against existing data.

### 6.6.1 General Economic Considerations

Because the various aforementioned economic models have been established at different points in time, inflation must be considered for consistency purposes. Using data from the U.S. Department of Labor, the cumulative Consumer Price Index (CPI) can be calculated exactly from 1914 to 2014 [92]. A model needs to be developed for projections after 2015. To establish this model, a linear function is used to approximate inflation. The coefficients of this model have been evaluated by creating a linear surrogate model with data from 1975 to 2014. This linear model is displayed in Figure 80 and represented in Equation 126, where $i_r(y)$ is the yearly average CPI relative to the baseline year 1914 and $y$ the fiscal year.

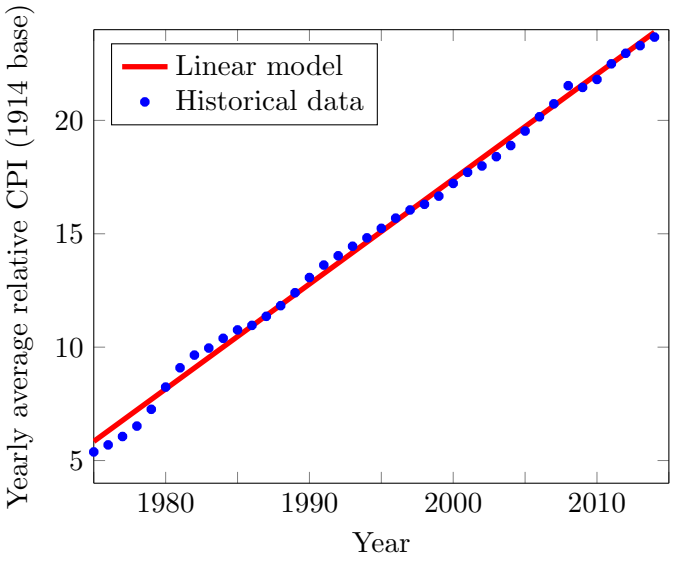



**Figure 80:** Yearly average relative CPI

The result of this regression is presented in Equation 126, where $i_r(y)$ is the yearly average CPI relative to the baseline year 1914 and $y$ the fiscal year.

$$i_r(y) = 0.4627y - 907.99 \tag{126}$$

The $R^2$ related to this equation is 0.9964, which is deemed accurate. By using this equation, it is assumed that the linear trend will remain constant in the future.

### 6.6.2 Architecture of the Economic Module

Following the decomposition suggested by Goehlich [176], life-cycle costs are divided into three main categories: development costs, manufacturing costs, and operating costs. The components of each of these categories are described in this section.

#### *6.6.2.1 Development Cost*

The development cost is a non-recurring cost that includes testing, fabrication rigs, and tools. For this research, its estimation is decomposed into two sub-categories: airframe and rocket engines. It is assumed that jet engines have already been developed for other applications so no development cost is associated with them.

#### *6.6.2.2 Manufacturing Cost*

The manufacturing cost is a recurring cost that includes prototype manufacturing and production costs. The manufacturing cost is decomposed into three sub-categories: airframe, rocket engines, and jet engines. The manufacturing cost of an item tends to decrease as the number of items produced increases. This effect called the "learning curve effect" has two roots. First, the production setup costs are distributed over more units. Consequently, its contribution to the unit cost is amortized. Then, there is a cost reduction due to improvements in production methods and better management. The learning curve is described by a percentage of learning $LC$, which usually varies between 0.85 and 0.95. Based on this factor, the manufacturing cost of the $N^{th}$ unit $C_M$ is defined in Equation 127, in which $C_1$ corresponds to the first unit cost. In this research, the percentage of learning is fixed and equal to 0.9.

$$C_M = C_1.N^{\frac{\ln LC}{\ln 2}} \tag{127}$$

#### *6.6.2.3 Operating Costs*

Operating costs are recurring costs that can be divided into several subcategories, as described below:

- DOCs: costs that are directly related to launch operations.

  - Variable DOCs: costs that are dependent on the vehicle's utilization.
    * Pre-launch operating cost: includes ground transportation, vehicle assembly, checkout, fueling, and launch preparations.
    * Launch operating cost: includes the communication system as well as the personnel and software efforts of the mission control center.
    * Propellant cost: includes propellants for all engines used by the vehicle.
    * Launch site cost: includes launch site administration, facilities maintenance, range stations, and safety provisions.
    * Vehicle insurance cost: considers the abort rate (launches that require a re-launch that is not paid by the customer) and the vehicle loss rate.
    * Vehicle amortization cost: used to spread out the capital expenses over the entire life of a system.
    * Transportation cost: includes emergency landing sites and ferry flights.
    * Maintenance cost: efforts needed to keep the vehicle at the same status as a newly built system.
  - Fixed DOCs: costs that are not dependent on the vehicle's utilization.
    * Development amortization cost: linear distribution of the amortization cost over all vehicles and all launches.
    * Financing cost: originates from a loan at the beginning of the operation phase.
    * Product improvement cost: due to reliability, maintenance, and security improvements.
    * Disposal cost: costs of scrapping vehicles and ground facilities and dismissing employees.

- IOCs: costs that are not directly related to launch operations and include marketing, customer relations, system management, spares storage, and pilot training.

The following section describes the architecture of the economic module and how the aforementioned costs are integrated into this module.

### 6.6.3 Architecture

The only cost category that can be calculated without inputting any other cost information is the manufacturing cost. Based on the manufacturing cost outputs and some of the design variables, the development cost can be determined and used to calculate the operating costs. In addition, since the insurance cost depends on the Total Operating Cost (TOC), its contribution is added after all other components. This helps avoid a computational loop and consequently speed up the overall optimization process. The general architecture is displayed in Figure 81, with the corresponding Matlab code being provided in Appendix D.2.3.

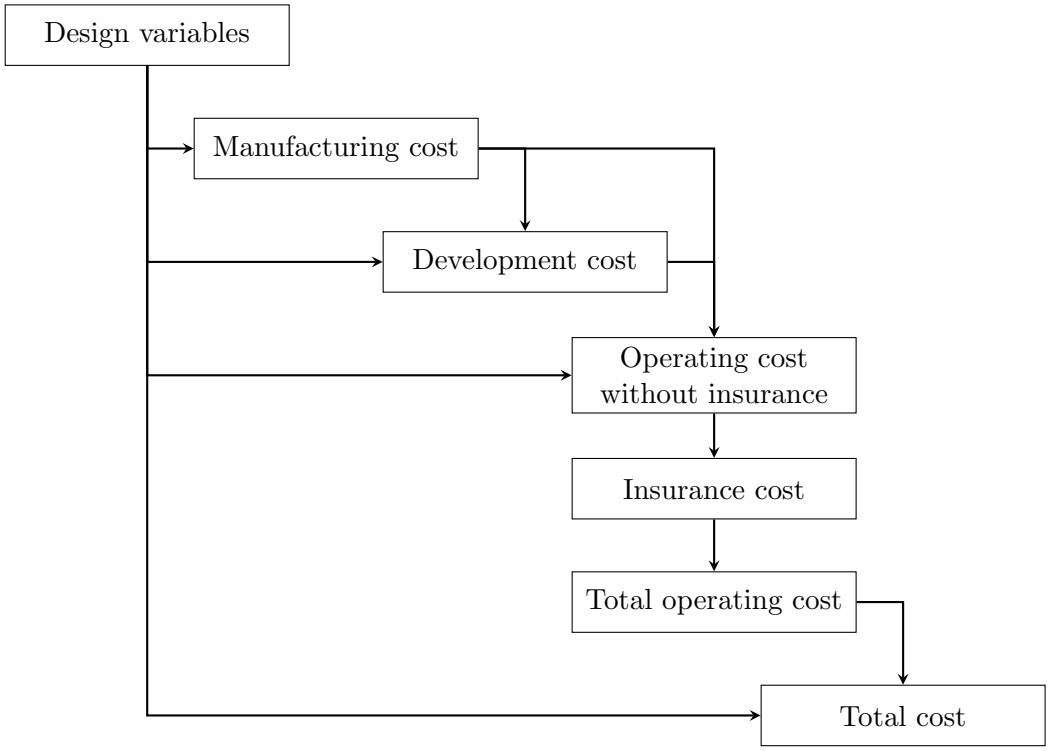



**Figure 81:** Architecture of the economic module

Based on the aforementioned architecture of the cost module, the following sections discuss the development of the various cost models.

### 6.6.4 Manufacturing Cost

As discussed in Section 3.3.6, Goehlich's CERs can be used to evaluate all relevant manufacturing costs except the ones related to the propulsion systems [176]. Hence, this section is organized around the following subsystems: airframe, rocket engines, and jet engines.

#### *6.6.4.1 Airframe*

Goehlich's model has been specifically developed for suborbital vehicles and can be used for single stage, first and second stage ballistic or winged vehicles [176]. The CERs rely on a series of assessment factors. In general, the lower the value, the lower the expenditure. For this research, the values of these factors remain constant and correspond to average values, as suggested by Goehlich [176]. The list of the different factors with the selected values and all the CERs issued from Suborb-TransCost are presented in Appendix B.5, from Equation 268 to Equation 284 [176].

#### *6.6.4.2 Rocket Engine*

**Solid rocket engine cost**

Graver et al. [182] developed CERs for the manufacturing cost of solid rocket engines based on the cumulative average cost at total production quantity $CAC(Q)$. The first unit cost of the rocket engine $C_{1,S}$ is then determined using Equation 128, where $LC$ is the learning curve factor and $Q$ the total production quantity.

$$C_{1,S} = \frac{CAC(Q)}{Q^{\frac{\ln LC}{\ln 2}}} \tag{128}$$

Different CERs based on either the Total Impulse ($TI$), the engine mass or the nozzle mass were developed. An analysis conducted by Graver [182] shows that the CER relying on $TI$ is the most suitable because of its accuracy and its validity in the entire design space. The corresponding relationship is presented in Equation 129 [182]. In this equation, $D_1$ and $D_2$ are two constants that are material-dependent. For the purpose of this research, $D_1$ and $D_2$ are set to 0 (assuming neither kevlar nor titanium is used). $N_N$ corresponds to the number

of nozzles and $Q$ the total produced quantity.

$$CAC(Q) = 77.595Q^{-0.3597}TI^{0.5081}N_N^{0.6116}1.4433^{D_1}1.2939^{D_2} \tag{129}$$

**Liquid rocket engine cost**

Friedland et al. [320] have developed CERs for calculating the first unit cost of liquid rocket engines. While different models are provided, the most accurate and the most stable one relies on three input variables: the engine dry weight $W_{liq}$, its vacuum thrust $T_R$, and its mass flow rate $\dot{m}$. The corresponding CER (Equation 130) was developed around a database of around 20 engines. The correlation coefficient is $R^2 = 0.89$ and the standard error of the logarithm is $\sigma = 0.244$.

$$C_{1,L} = 374W_{liq}^{1.718}T_R^{-0.043}\dot{m}^{-0.827} \tag{130}$$

**Hybrid rocket engine cost**

Since no CERs exist for hybrid engines, a more physics-based approach must be implemented. Hybrid engines can be broken down into 4 main parts: a typical solid engine, a pressurization tank, an oxidizer tank, and a feed system. As such, the first unit cost of a hybrid engine $C_{1,H}$ can be calculated using Equation 131, where $C'_{1,S}$ is the cost of the solid sub-engine, $C_{1,tkP}$ the cost of the pressurization tank, $C_{1,tkO}$ the cost of the oxidizer tank, and $C_{1,fs}$ the cost of the feed system. The models used for each of these components are presented below:

$$C_{1,H} = C'_{1,S} + C_{1,tkP} + C_{1,tkO} + C_{1,fs} \tag{131}$$

- Solid sub-engine: Graver and Morrison [182] provide three different CERs for solid engines based on the total impulse, the solid engine weight, and the nozzle weight. According to Zandbergen [494], the weight of a solid engine can be directly linked to its propellant weight, which can be calculated with the overall propellant weight and the optimum oxidizer-to-fuel ratio known from the performance evaluation module discussed in Section 6.3. The first unit cost is then determined using Equation 132,

where $W_{sol}$ is the solid engine weight including the propellant.

$$CAC(Q) = 22.045 Q^{-0.3387} W_{sol}^{0.5126} N_N^{0.6167} 1.6680^{D_1} 1.3867^{D_2} \tag{132}$$

- Tanks: Brown et al. [56] from NASA provide CERs for both recurring $C_{t,r}$ and non-recurring $C_{t,nr}$ manufacturing costs of propellant tanks that include material, tool design labor, tool material, tool manufacturing labor, production labor, manufacturing management, and quality control labor. As such, the first unit cost of each tank $C_{1,tk}$ is calculated using the sum of the recurring and non-recurring costs. CERs are developed around the area of the tank $S_t$ defined in Equation 133, where $R_t$ is defined as the radius of the tank and $L_t$ its length.

$$S_t = 2\pi R_t L_t + 4\pi R_t^2 \tag{133}$$

  Thus, the recurring cost $C_{t,r}$ is defined by $C_{t,r} = 328\ S_t$, while the non-recurring cost $C_{t,nr}$ is defined by $C_{t,nr} = 2,660\ S_t$.

- Feed system: Meisl [291] provides a manufacturing cost breakdown for reusable liquid engines. The propellant feed system represents 11% of the total liquid engine cost. As such, the feed system cost can be evaluated based on the cost of the liquid sub-engine $C'_{1,L}$ as a function of the engine weight $W_{liq}$ using the CER developed by Friedland et al. [320]. $W_{liq}$ can be determined using the process described in Section 6.3.1.4. The final first unit cost estimation equation for the feed system is provided in Equation 134.

$$C_{1,fs} = 0.11 C'_{1,L} = 398.2 W_{liq}^{0.657} \tag{134}$$

#### *6.6.4.3 Jet Engine Cost*

Suborb-TransCost suggests a CER based on only one design variable to evaluate jet engine costs: the engine weight. Hence, a better model is needed to be able to perform trade-off analyses. Younoussi et al. [489] provide a model based on five variables including the learning curve factor. It is also assumed that the engine is not developed especially for suborbital vehicles. They provide Equation 135 to evaluate the engine manufacturing cost

$C_{M,J}$, where $T_4$ is the turbine inlet temperature, $LC$ the learning curve factor, $ab$ a binary variable (1 if the engine is an afterburning engine and 0 if it is not), and $W_{jet}$ the engine weight.

$$\ln C_{M,J} = -10.40 - 8.55 \ln LC + 0.482ab + 1.162 \ln T_4 + 0.261 W_{jet} \tag{135}$$

### 6.6.5 Development Cost

The development cost for airframe is provided by Goehlich and follows CERs similar to the ones for the manufacturing cost. They are also provided in Appendix B.5.

According to Morrison [185], the development cost of solid and liquid rocket engines only depends on the number of prototypes $Proto$ and the cumulative average cost of 150 engines $CAC_{150}$, as expressed in Equation 136.

$$C_D = 52.947\, CAC_{150}^{0.939} Proto^{0.618} \tag{136}$$

$CAC_{150}$ can be calculated using the first unit cost determined in the previous section and the learning curve effect, as presented in Equation 127. The previous section also showed that the manufacturing cost of hybrid engines can be calculated using existing relationships developed for solid and liquid engines. As a result, it is assumed that Equation 136 is also valid for hybrid engines.

### 6.6.6 Operating Costs

The operating costs related to the airframe and to the rocket engine maintenance are provided by Goehlich. The CERs are presented in Appendix B.5. The cost of the various propellants is provided by the Defense Logistics Agency [111] and Killian et al. [234] in 2015 U.S. $ per ton: $O_2$ (1,761), $H_2$ (30,764), RP1 (21,755), $N_2O_4$ (226,000), MMH (185,120), Aerozine50 (185,120), $N_2O$ (2,773), and solid (1,110).

The jet engine maintenance cost can be estimated based on the study performed by Liebeck et al. [257]. In their study, the authors divide the engine maintenance cost into three different categories: labor, material, and applied maintenance burden. Equation 137 provides the final CER for estimating total maintenance cost per trip $C_{M,jet}$, where $T_j$ is

the available thrust at sea level, $t_t$ the trip time in hours, and $n_j$ the number of jet engines.

$$C_{M,jet} = \left(75\left(0.645 + \frac{0.05T_j}{10^4}\right)\left(0.566 + \frac{0.434}{t_t}\right) + \left(25 + \frac{0.05T_j}{10^4}\right)\left(0.62 + \frac{0.38}{t_t}\right)\right) t_t.n_J \tag{137}$$

The jet engine fuel cost is calculated based on the jet fuel consumption during the trip and its cost. According to the U.S. Energy Information Administration, the average jet fuel price over the next 10 years will be \$24/million Btu or around \$3 per gallon [95].

The flight crew cost can be estimated using the Association of European Airlines (AEA)'s approach, which consists in allocating \$245 (1989 U.S.\$) per flight crew member and per block hour [11].

### 6.6.7 Validation of the Economic Module

Validating the life-cycle cost estimation model is a challenging task, due to the fact that economic data for aerospace and defense systems are often proprietary and thus very hard to get access to.

#### *6.6.7.1 Airframe Cost*

Since there is no active suborbital vehicle, each cost component must be checked individually with different known systems and scenarios.

**Table 32:** Validation of the airframe cost model [176]

| **Cost components** | **Variables** | **Results (M\$)** | **Deviations** |
|---|---|---|---|
| $C_{D,B}$ (BETA) | $M_0$=450Mg | 10,342 | -10% |
| $C_{D,W}$ (Concorde) | $M_W$=65Mg, $v_w$=2.2 | 11,489 | +9% |
| $C_{V,W}$ (X-15) | $M_W$=7Mg | 67 | +7% |
| $C_{pl}$ (STS) | $M_0$=2,000Mg, $L$=5, VTO | 157/launch | +15% |
| $C_{pl}$ (Space Shuttle Orbiter) | $L$=12, $T_m$=240h, $n_{Crew}$=7 | 8.57/launch | +17% |
| $C_{maint}$ (Space Shuttle Orbiter) | $C_{V,W}$=\$4.26×10$^9$, $C_{V,R}$=\$3.73×10$^8$ | 17.6/launch | 0% |

Hence, the model has been checked using a sample of six data points provided by Koelle as described in Table 32 with a reference year equal to 2000 [176]. It provides an accuracy of about ±17%, which is highly accurate, especially for cost estimation. This accuracy is similar to the one obtained with the original Transcost model for spacecraft.

#### *6.6.7.2 Rocket Engine Cost*

Only data for ten engines have been found in the literature. They represent five development costs and seven manufacturing costs (four solid engines and six liquid engines). Tables 33 and 34 provide the input parameters used for the validation of the rocket engine cost estimation modules. The year mentioned corresponds to the year at which the cost has been evaluated.

**Table 33:** Input parameters for the validation of the solid engine cost estimation

| **Engine** | **Propellant mass (lb)** | **Isp (s)** | **Number of nozzles** | **Number of prototypes** | **Quantity** | **Year** |
|---|---|---|---|---|---|---|
| Castor 120 [131] | 108,097 | 229 | 1 | - | 18 | 1997 |
| Orion 50 [133] | 6,600 | 292 | 1 | - | 52 | 1999 |
| Orion 50S [134] | 26,782 | 294 | 1 | - | 24 | 1999 |
| GEM 40 [132] | 25,801 | 274 | 1 | - | 1,631 | 1999 |

Based on this comparison, a mean relative error of 19% was found. This is better than the average error claimed by TransCost [240] and other traditional cost estimation programs.

Even though no cost information has been found for hybrid engines, the model is assumed to be validated as it originates from a process that follows a rigorous approach validated step by step. In addition, Experiment 3.2 is performed. The proposed methodology is used to evaluate the life-cycle costs of each type of rocket propellant. For validation purposes, the methodology is used to calculate the life-cycle cost required for the development, production, and operations of a single rocket engine that performs 100 flights. It is required that the engine fits into a fuselage with a two-meter diameter, while providing a

**Table 34:** Input parameters for the validation of the liquid engine cost estimation

| Engine | Weight (lb) | Isp (s) | Thrust (lbf) | Number of prototypes | Manufacturing cost (Year) | RDT&E cost (Year) |
|---|---|---|---|---|---|---|
| J-2 [185] | 3,942 | 421 | 232,250 | 38 | $3.7 M (1989) | $168 M (1963) |
| RL10 [185] | 298 | 465 | 14,994 | - | $7.0 M (1989) | - |
| Agena B [185] | 1,911 | 285 | 15,999 | - | $8.5 M (1989) | - |
| LE-7 [493] | 3,900 | 446 | 242,000 | 37 | - | $781 M (1991) |
| LE-5 [493] | 540 | 450 | 23,155 | 37 | - | $108 M (1986) |
| F-1 [185, 202] | 18,498 | 304 | 1,516,898 | 56 | $9.3 M (1989) | $450 M (1966) |
| H-1 [202, 237] | 2,100 | 289 | 205,000 | 10 | - | $17.7 M (1962) |

1,000-kN thrust during two minutes. In order to identify trends, a DoE of 1,000 points is generated on both the chamber pressure and the nozzle expansion ratio following a Latin Hypercube model. The distribution of costs are grouped by type of rocket engine and presented in Table 35.

**Table 35:** Validation of the hybrid engine cost estimation model

| Propellant type | Mean (U.S. $ million) | Standard deviation (U.S. $ million) |
|---|---|---|
| Solid | 795 | 0.22 |
| Liquid | 31,280 | 16,450.21 |
| Hybrid | 1,173 | 134.62 |

As shown in Table 35, hybrid engines are more expensive than solid engines but less expensive than liquid engines. This validates Hypothesis 3.2:

**VALIDATION HYPOTHESIS 3.2: IF a hybrid rocket engine is physically decomposed into a liquid engine without its nozzle and a conventional solid engine THEN its life-cycle costs can be predicted using existing models developed for liquid and solid engines.**

#### *6.6.7.3 Jet Engine Cost*

The application of the CER presented in Equation 135 for estimating the manufacturing cost of 24 engines provides a $R^2$ of 0.97. In addition, the residual vs. predicted plot is provided in Figure 82 and shows a good correlation, without strong patterns or large discrepancies in the distribution [489].

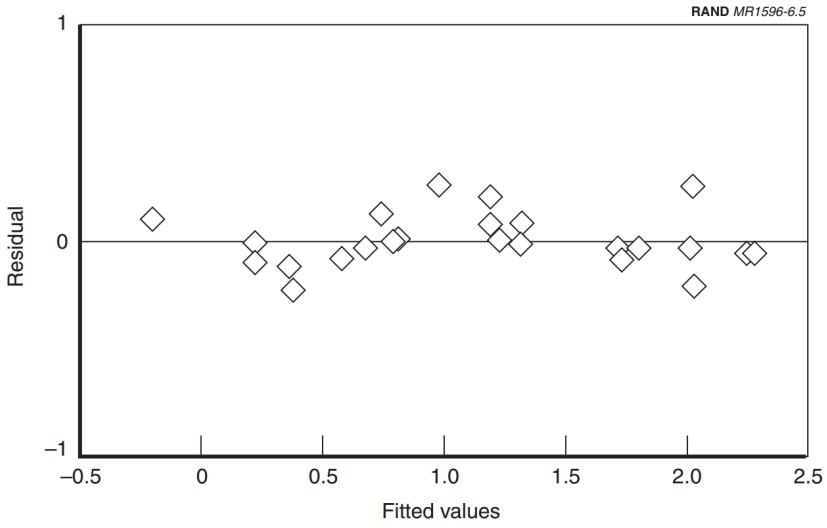



**Figure 82:** Residual vs. calculated production cost [489]

Sections 6.2 to 6.6 have discussed the various modeling techniques used to evaluate both the flying and the economic performance of suborbital vehicles. The following section provides the approach for risk and safety analysis used in this research.

## *6.7 Safety Analysis*

As discussed in Section 3.3.7, comparing and selecting architectures cannot be accomplished without safety considerations. Safety is defined by the United States Air Force (USAF) as the "freedom from those conditions which can cause injury or death to personnel, damage to or loss of equipment or property" [8]. Hence, the safety level of a vehicle is directly related to its risk level, which needs to be reduced in order to improve the vehicle's safety. A modified FTA, which is a top-down or deductive approach to reliability modeling, is used to evaluate the risk level. It can be calculated using the general formulation presented in Equation 138, which includes both the failure occurrence and the

failure severity. Based on this definition, the smaller the risk value, the safer the vehicle.

$$Risk = Occurrence \times Severity \tag{138}$$

Moreover, for this research, the FTA is decomposed into three levels:

- Top level: the vehicle architecture.
- Medium level: the different vehicle features.
- Basis level: the different options chosen to fulfill each feature.

Suborbital vehicles can be decomposed into five main features impacting the safety, which represent the medium level: rocket propulsion, jet propulsion, launch method, landing method, and number of pilots. Each of these features has multiple options, which populate the basis level. For example, the rocket propulsion can be provided by solid, liquid or hybrid engines.

Based on this general approach to risk calculation, the architecture risk level is defined in Equation 139, where $\alpha_i$ represents the failure rate of the $i^{th}$ feature and $\gamma_i$ the failure severity of the $i^{th}$ feature.

$$\text{Risk level} = \sum_i (\alpha_i \times \gamma_i) \tag{139}$$

The coefficient $\alpha$ is defined as the probability that a given feature fails compared to the others. The coefficients $\alpha_i$ are determined historically by identifying the source of all previous failures or accidents in both the space and aeronautic fields. In addition, it is assumed that the feature failure rate does not depend on the selected option. This is a simplified assumption deemed acceptable for calculations at the conceptual design phase since no quantified information is available. The two coefficients $\alpha$ and $\gamma$ are consequently considered to be independent. The results of this analysis are presented in Table 36.

In order to evaluate the failure severity of each feature, the coefficient $\gamma$ is introduced based on the technical characteristic of each option. Its value varies between 0 to 10, where 0 is the safest and 10 the riskiest. In order to account for both discrete and continuous

variables of a given feature, the value of $\gamma$ is decomposed into two components: a baseline value $\beta$ that depends on discrete parameters and a variable term that depends on continuous parameters.

**Table 36:** Failure occurrence of each feature [43, 76, 376]

| **Vehicle architecture features** | $\boldsymbol{\alpha}$ |
|---|---|
| Rocket propulsion | 75.95% |
| Jet propulsion | 1.27% |
| Launch method | 7.38% |
| Landing method | 10.44% |
| Number of pilots | 4.96% |

For each feature, the calculation of $\gamma$ is presented in Equation 140, where $\beta_k$ is the baseline value of the failure severity of the k$^{\text{th}}$ feature, $x_j$ the j$^{\text{th}}$ variable parameter that impacts the failure severity and $\eta_j$ the intensity of its impact in percentage.

$$\gamma = \beta_k \left( 1 + \sum_j \eta_j x_j \right) \tag{140}$$

The following sections aim at determining the values of $\gamma$ for all the features described in Table 36.

### 6.7.1 Rocket Propulsion

#### *6.7.1.1 Liquid Rocket Propulsion Systems*

Liquid Rocket Propulsion Systems (LRPSs) are the most widely used type of rocket propulsion. In this research, only bi-propellants LRPSs are considered and are decomposed into two categories: tank pressure-fed and turbopump pressure-fed LRPSs [211, 215, 216, 305, 376].

For the safety assessment of bi-propellant LRPSs, no distinction is made between the two categories. Indeed, while turbopump pressure-fed LRPSs are more complex due to a higher number of parts, tank pressured-fed LRPSs have highly pressurized propellant tanks.

While many sources of risk exist for LRPSs, the main one is related to pressure propellant leaks. Indeed, leakage of propellant during the flight can lead to dramatic deterioration in the vehicle performance and consequently compromise both the vehicle and the mission's integrity. On the ground, propellant leakages can also result in important damages to the launch pad and hurt the maintenance team since many of the storable chemical propellants are toxic and/or carcinogenic. In addition, contrary to rockets, suborbital vehicles have to land with their LRPSs. Consequently, LRPSs need to be able to withstand the constraints of the landing phases and have a planned post-life depressurization. Moreover, during the microgravity phase, LRPSs residual high pressure can leak into a low pressure subsystem that was not designed to hold over-pressure. As a result, low-pressure subsystems can explode along with the residual propellants.

Operating conditions for LRPSs are also very stringent. A failure of the cooling and heating systems can have dramatic consequences. Indeed, the propellant can freeze or react with hot surfaces causing damages to the vehicle and compromising the mission.

The environment in which LRPSs operate can also cause damages. Indeed, LRPSs are exposed to aggressive environmental conditions such as humidity, which could lead to destructive effects. Most fluid propulsion system components are vulnerable to internal leakage caused by particulate contamination. Hence, LRPSs have to be manufactured, assembled, tested, and stored in a dedicated clean room.

Finally, the risk of unwanted ignition is considered to be low. Indeed, the probability that both fuel and oxidizer valves are involuntarily opened at the same time is extremely low and a redundancy on vital systems is present for each valve.

According to the previous studies, LRPSs are assumed to have a baseline failure severity level of 5 out of 10. The risk level of LRPSs also depends on the propellant used. In this research, three liquid propellant mixtures are considered: LOx/$LH_2$, LOx/RP-1, and LOx/Hypergolic propellant. RP-1 is a highly refined kerosene, which produces a lower specific impulse than $LH_2$ but is safer. $LH_2$ is very cryogenic and has a very low density, so LRPSs using LOx/$LH_2$ and LOx/RP-1 mixtures have a baseline failure severity level of 5 and 4.5, respectively. Hypergolic propellants are used by simpler system because they do

not need ignition. However, the ratio between hypergolic propellants and LOx inside the combustion chamber has to be extremely precise. Indeed, a bad mixing directly results in a fatal failure for the vehicle. For that reason, LRPSs using LOx/Hypergolic propellants have a baseline failure severity level of 5.5 out of 10.

#### *6.7.1.2 Solid Rocket Motors*

Hazards due to Solid Rocket Motors (SRMs) are widely known mainly because to the tragic fate of the Space Shuttle Challenger back in 1986 [310]. Because SRMs use solid propellant grains composed of both the oxidizer and the fuel, the risk of unwanted ignition is high. Combined with the inability to stop the combustion process, a failure in SRMs ultimately results in fatal failures where the integrity of both the vehicle and the crew are compromised [216, 305].

Electrostatic discharge, radio frequency, electromagnetic field, and lightning can also result in unwanted ignition. To prevent electrostatic discharge, every person, equipment, storage, and transport around SRMs have to be grounded. Every wireless equipment or cell phone used by the team near SRMs has to be turned off or shielded to prevent inadvertent firing caused by radio waves and electromagnetic fields. Lightning protection on SRM storage buildings or launch pads is also necessary to avoid ignition risks.

Moreover, propellant grain cracks or high internal deformation stresses can lead to explosion. Indeed, pieces of propellant can break due to inappropriate shocks, causing nozzle blockage, increased burning surface area, and debonding between the propellant and the liner. The solid propellant system should be embedded into a pressurized container during transport.

Temperature is also a critical parameter impacting the safety of SRMs. Indeed, it changes its modulus of elasticity and the propellant becomes extremely breakable. A shock or an excessive load at low temperatures can have a damaging effect on the integrity of the propellant grain. Debonds can result in uncontrolled burning or local overheating. Low temperatures weaken O-rings and seals as it happened during the Challenger accident. These under-cooling risks are higher than the overheating ones. Storage facilities of solid

propellant systems must be controlled in terms of temperature and humidity.

Based on the previous analysis, SRMs are considered to be the most dangerous type of rocket engines and are assumed to have a baseline failure severity level of 8 out of 10.

#### *6.7.1.3 Hybrid Rocket Propulsion Systems*

Hybrid Rocket Propulsion Systems (HRPSs) use propellants in two different states: a liquid oxidizer and a solid fuel. While the opposite configuration might exist, only the first one is considered. In terms of safety, the risks related to HRPSs are not the sum of the ones from SRMs and LRPSs [216, 376].

According to the research performed by the U.S. Department of Transportation on hazard analysis of commercial space transportation [323], HRPSs' inadvertent ignition is nearly impossible due to the separation of the oxidizer and the fuel. Hence, there is no explosion or detonation risk during HRPSs manufacturing, storage, and operation.

The risk level of HRPSs is similar to the one of jet engines. There is almost zero probability of catastrophic failure. Nevertheless, in some cases, thrust may not be produced. However, HRPSs can be shut down and restarted. Therefore, in case of failure, HRPSs would be shut down and an emergency landing would be performed.

According to the previous observations, HRPSs are assumed to have a baseline failure severity level of 3 out of 10. In addition, the risk level of HRPSs also depends on the propellant used. In this research, four propellant mixtures are considered for HRPSs: LOx/HTPB, LOx/Paraffin, $N_2O$/HTPB, and $N_2O$/Paraffin. Using paraffin avoids the need for complex HTPB cross-section, casting, and multi-point injector in order to increase both the reliability and the performance of the engines. When $N_2O$ is used in HRPSs, the injection is less stable and more complex to predict and control. Hence, $N_2O$ is considered to be riskier than LOx. Based on these observations, HRPSs using LOx/HTPB and $N_2O$/Paraffin mixtures have a baseline failure severity level of 3, while LOx/Paraffin and $N_2O$/HTPB have a baseline failure severity level of 2 and 4, respectively.

#### *6.7.1.4 Conclusion*

Based on the previous analysis, the different options can be classified in terms of risk level, as displayed in Table 37 [361, 410].

**Table 37:** Failure severity coefficients for rocket propulsion options

| **Rocket engine types** | **Propellant mixtures used** | $\beta$ |
|---|---|---|
| **SRMs** | | 8 |
| **LRPSs** | $LOx/LH_2$ | 5 |
| | LOx/RP-1 | 4.5 |
| **HRPSs** | $N_2O$/HTPB | 4 |
| | $N_2O$/Paraffin | 3 |
| | LOx/HTPB | 3 |
| | LOx/Paraffin | 2 |

The rocket propulsion failure severity level also depends on the chamber pressure $p_c$. Indeed, the higher $p_c$, the higher the failure severity. The calculation of $\gamma$ is presented in Equation 141, where $C_{p_c}$ is a coefficient proportional to $p_c$ that varies between -10% and +10% around $\beta$.

$$\gamma = \beta + C_{p_c} \tag{141}$$

Figure 83 illustrates the behavior of the risk level of the different types of rocket engines. A representative propellant mixture has been chosen for each type of propellant: $LOx/LH_2$ for liquid engines and $N_2O$/Paraffin for hybrid engines. The figure shows the impact of the type of propellant, chamber pressure, and thrust on risk level.

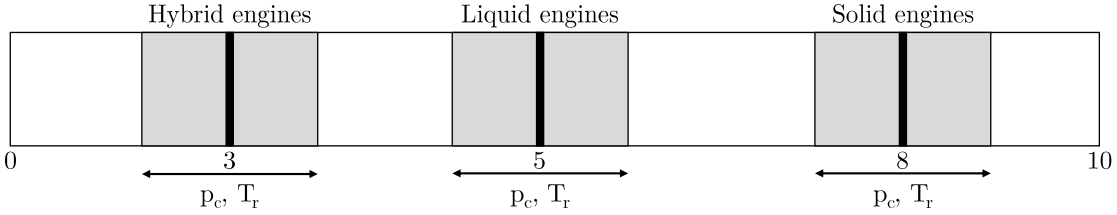



**Figure 83:** Behavior of the risk level of rocket engines

### 6.7.2 Jet Propulsion

The risk level of the overall jet propulsion system mainly depends on the type of jet engine and the number of jet engines. These two criteria are discussed in the following sections.

#### *6.7.2.1 Type of Jet Engine*

During missions, jet engines are exposed to multiple risks. One of the highest hazards is the risk of ingestion of foreign objects such as birds, sand, stones, and ice. The lack of containment of broken blading can result in uncontrollable fires, which prompt the engine disintegration [292, 444].

The jet engine risk level is also related to airflow disruption with flameouts, which can result in severe power losses and minor blade damages.

The use of afterburners consists in injecting kerosene into the outlet nozzle, which immediately inflames with the turbine hot gas. While they provide an additional thrust to the vehicle, the fuel consumption highly increases and the combustion efficiency drastically decreases. Afterburners can be implemented on both turbojet and turbofan engines. The additional combustion also highly increases the outlet nozzle temperature, creating additional structural and mechanical constraints that cannot be sustained too long by the nozzle. Hence, afterburners can only be turned on during a small period of time. The use of afterburners consequently increases the risk of failure due to the extra combustion in the outlet nozzle.

#### *6.7.2.2 Number of Jet Engines*

The higher the number of jet engines, the lower the risk of crash. Indeed, if one of the jet engines fails, pilots can shut it down and perform a safe landing using the other engine(s) on a diversion runway.

Moreover, having a vehicle without jet engine highly increases the risk of catastrophic failure for two main reasons. First, the vehicle is only powered by rocket engines, which have a higher risk of failure than jet engines. Then, if there is a loss of power from the

rocket engine, the vehicle cannot use its jet engines to perform a safe emergency landing.

#### *6.7.2.3 Conclusion*

Based on the aforementioned observations, turbojets and turbofans are assumed to have the same baseline failure severity level: 2. Adding afterburners increases the risk, such that augmented turbojets and augmented turbofans have a baseline failure severity level of 5. Finally, having no jet engine is the most dangerous option and has a baseline failure severity level of 9. Table 38 summarizes these values.

**Table 38:** Failure severity coefficient for jet engines

| **Type of jet engine** | $\boldsymbol{\beta}$ |
|---|---|
| Turbojet | 2 |
| Turbofan | |
| Augmented turbojet | 5 |
| Augmented turbofan | |
| No jet engine | 9 |

The jet engine safety level also depends on the number of jet engines, as explained in the risk analysis. Hence, the calculation of $\gamma$ presented in Equation 142 depends on $C_{n,jet}$, which is a coefficient proportional to the number of jet engines and varies between -10% and +10% around $\beta$.

$$\gamma = \beta + C_{n,jet} \tag{142}$$

### 6.7.3 Launch Methods

Three take-off modes exist: vertical take-off, horizontal take-off, and air launched. Looking back at previous space missions, manned space flights have always been launched vertically. However, the presented concepts for suborbital vehicles have been defined around the three different types of launch methods. The risk level of each of these launch methods is discussed in the following sections [76, 376].

#### *6.7.3.1 Vertical Take-Off*

The vertical take-off mode is the most widely used and most known launching technique for spaceflights. The vehicle is launched from a small-size dedicated launch pad. Vertical take-off vehicles have a simple design and the longitudinal control of the center of mass is not necessary. The launch procedure that requires the vehicle to stand on its launch pad represents the main hazard of the vertical launch method. Indeed, during this time, the vehicle has to face environmental constraints such as rain, snow, temperature, wind, lightning, etc. The refueling operation can also be dangerous for the flight crew, the vehicle, and the ground maintenance teams. Finally, the engine ignition and the vehicle release from the pad also represent a non-negligible risk.

Based on these observations, the vertical take-off method appears to be relatively safe. Hence, it is assumed that its baseline failure severity level is equal to 4 our of 10.

#### *6.7.3.2 Horizontal Take-Off*

Horizontal take-offs impose strong constraints to the vehicle architecture. To horizontally take off from a runway, wings and landing gears are needed. Hence, this results in a more complex architecture for the vehicle compared to vertical take-off or air launched vehicles. In addition, fuselage and wings have to be designed stiff enough to withstand turbulent air, supersonic flights, and the pull-up maneuver after take-off. For a given material, these constraints increase the vehicle structural requirements and consequently its empty mass. Moreover, to keep the vehicle controllable, it is necessary to maintain its center of mass in a narrow longitudinal range. Two types of horizontal take-offs can be identified based on the engines used to take off: jet engines or rocket engines. The take-off with jet engines is the safest since it is used by millions of aircraft every year and is well known. In addition, the rocket engine is only used at high altitudes, once jet engines are turned off, and during a shorter period. However, rocket-powered take-offs are riskier since they result in the generation of a huge amount of thrust at low altitude with a type of engines that is not commonly used for aircraft.

As a consequence, rocket-powered horizontal take-offs appear to be more dangerous

than vertical take-offs and are assumed to have a baseline failure severity level equal to 6. Take-offs performed with jet engines are assumed to be the safest and a baseline failure severity level of 2 is assumed.

The risk level also depends on the take-off velocity $V_{to}$. Hence, the higher the take-off speed, the riskier the take-off. The calculation of $\gamma$ is presented in Equation 143, where $C_{V_{to}}$ is a coefficient proportional to $V_{to}$ that varies between -10% and +10% around $\beta$.

$$\gamma = \beta + C_{V_{to}} \tag{143}$$

#### *6.7.3.3 Air Launch*

The air launch method consists in launching the vehicle from a dedicated spacecraft carrier from a given altitude. Hence, the mass of propellant needed for the suborbital vehicle can be reduced. However, the geometry is constrained by the carrier capacity. The absence of ground reflection and the low air density decrease the engine vibrations that can cause damages to the fuselage. Moreover, launching the vehicle in the air does not impose any constraints or requirements on the launching site as long as the spacecraft carrier can take off safely.

The vehicle and the aircraft can be considered as two different stages. As such, the most dangerous phase of the mission is the vehicle separation followed by the free fall engine ignition. The separation could create hazardous debris for the vehicle and the population underneath, such as cable or explosive bolt coming from the jettisoning system. Government and safety regulations may require that the vehicle be launched over empty spaces such as ocean or uninhabited territories.

If the free fall rocket engine ignition fails, the propellant tank has to be rapidly drained so that an emergency landing can be performed. Moreover, the free fall launch requires an extremely precise control of the longitudinal center of mass. Fuel and oxidizer tanks have to be divided into smaller ones with dedicated transfer systems. If the vehicle uses cryogenic propellants and is carried outside the spacecraft carrier, the propellant may boil off due to the sun radiation or convective heating from the air stream.

Based on the previous observations, air launch is considered as the most dangerous method and has a baseline failure severity level of 8.

#### *6.7.3.4 Conclusion*

According to the previous analysis, the risk levels of the different options are summarized in Table 39.

**Table 39:** Failure severity coefficient for launch methods

| **Launch methods** | $\beta$ |
|---|---|
| Jet-powered horizontal take-off | 2 |
| Vertical take-off | 4 |
| Rocket-powered horizontal take-off | 6 |
| Air launch | 8 |

### 6.7.4 Landing Methods

During a suborbital flight, landing is the most dangerous mission phase. Three landing modes are considered in this research: vertical landing using parachute, horizontal gliding landing, and conventional powered horizontal landing. Both orbital and suborbital manned flights have already landed horizontally such as the American Space Shuttle and the experimental planes X-1 and X-15 [68]. In the early era of space flights, vertical landing was also performed. Mercury, Gemini, and Apollo spacecraft landed in the water using three parachutes to reduce the impact speed. In contrary, Russians preferred the ground landing technique using a single parachute and a retrorocket to soften the landing [68].

#### *6.7.4.1 Vertical Landing using Parachute*

Vertical landing using parachute has often been used for the first spaceflights and has the advantage of not requiring a long runway. However, the main challenge of the vertical landing using parachutes is the attenuation of the touchdown impact for the crew. Indeed, as the vehicle only has one vertical structural load path, parachutes are used to slow down the descent velocity. The landing maneuver does not need to be performed by a highly trained human pilot or by a complex automated landing system, as it only requires the

parachutes to be opened at a specific altitude [376]. A parachute opening system has to be extremely reliable since a failure directly results in a dramatic crash and the crew's death, as it happened during the first Soyuz mission on April $24^{\text{th}}$ 1967 with the loss of the cosmonaut Vladimir Komarov [29]. Multiple parachutes can be deployed to ensure redundancy. The uncertainty of the landing point is relatively high so astronauts might land in a hostile environment. Furthermore, the area has to be large and flat to prevent any hazard of tilting back.

Water landing is a smooth method used by NASA. Because water dampens the impact, the weight of the parachute system can be reduced. However, this landing method requires an expensive and complex recovery process such as ships and helicopters. The vehicles also have to be carefully disassembled and repaired after every flight due to the contact with corrosive salty water. These constraints might become a critical problem for reusable vehicles with a high flight frequency.

Another technique, mostly used by Russians, consists in firing retrorockets before touchdown in order to reduce the descent velocity and the vehicle's impact. To perform this maneuver, the vehicle's heat shield has to be jettisoned, creating dangerous pieces of debris. Similar to parachutes, retrorockets have to be very reliable because in case of failure, the descent rate would be too high and the vehicle would be destroyed. Finally, retrorockets can also cause a ground fire.

Based on the previous discussion, the vertical landing method is considered as the safest recovery system and has a baseline failure severity level of 3.

#### *6.7.4.2 Conventional Horizontal Landing and Gliding*

Conventional and gliding landings are the most precise way of landing as the vehicle touchdown is performed on a dedicated runway at a precisely known location. It is the softest landing compared to the other landing methods, with a touchdown vertical velocity inferior to 1 m/s [345, 376]. However, the vehicle is more complex, leading to multiple safety concerns. Wings, landing gears, and control surfaces are necessary to modify the trajectory of the vehicle by controlling its pitch, roll, and yaw. In addition, having a highly qualified

pilot and/or a complex automated landing system is mandatory. If the vehicle crashes, the area impacted by the crash would be large compared to a vertical landing crash.

For the gliding method, the vehicle needs a high lift-to-drag ratio. The main risk is caused by the absence of engine since the vehicle only has one chance to safely land. Moreover, the trajectory correction range is smaller than for a conventional landing.

During a conventional landing, jet engines are turned on again to provide more controllability to the vehicle with respect to its longitudinal flight path. This allows for the landing to be aborted and a go-around maneuver performed. Jet engines present the main source of risk as the propellant is stored in the tanks during the re-entry phase and can explode. A jet engine failure ignition could also force the pilot to perform an emergency gliding landing.

Based on the previous study, horizontal landings are definitively more dangerous due to the complexity of the systems needed. Conventional horizontal landings have a baseline failure severity level of 5 while gliding is considered as the most dangerous method, with a baseline failure severity level of 9.

The risk level also depends on the landing velocity $V_{la}$. Indeed, the higher $V_{la}$, the higher the risk level. The calculation of $\gamma$ is presented in Equation 144, where $C_{V_{la}}$ is a coefficient proportional to $V_{la}$ that varies between -10% and +10% around $\beta$.

$$\gamma = \beta + C_{V_{la}} \tag{144}$$

#### *6.7.4.3 Conclusion*

Based on this discussion, the baseline failure severity level of the different landing methods are presented in Table 40.

**Table 40:** Failure severity coefficient for landing methods

| **Landing methods** | $\boldsymbol{\beta}$ |
|---|---|
| Vertical landing | 3 |
| Horizontal conventional landing | 5 |
| Horizontal gliding landing | 9 |

### 6.7.5 Pilots

Despite recent developments in complex automated pilot systems and the number of automated mission phases, the presence of a pilot is considered to be safer. Indeed, during an emergency situation, highly trained and qualified pilot(s) can identify, analyze, and handle failures in order to perform a safe and emergency landing.

As the number of pilots in the cockpit increases, it becomes easier and faster to treat failures since the individual workload is reduced. Nevertheless, this requires pilots to be highly trained in order to avoid misunderstandings. Table 41 displays the failure severity coefficient selected for each option.

**Table 41:** Failure severity coefficient for the number of pilots

| **Number of pilots** | $\beta$ |
|---|---|
| No pilot | 8 |
| One pilot | 5 |
| Two pilots | 2 |

### 6.7.6 Passengers

In case of failure, the number of passengers is directly related to the number of human losses. Hence, the more passengers on board, the higher the severity of the failure. However, because this does not represent a technical risk, it will not be taken into account in this research.

### 6.7.7 Conclusion

Figure 84 displays the final FTA developed in this research. It includes the failure occurrence rate of each feature and provides all alternatives that have been considered when calculating the safety level of the vehicle. The proposed approach is implemented into Matlab, as detailed in Appendix D.2.6.

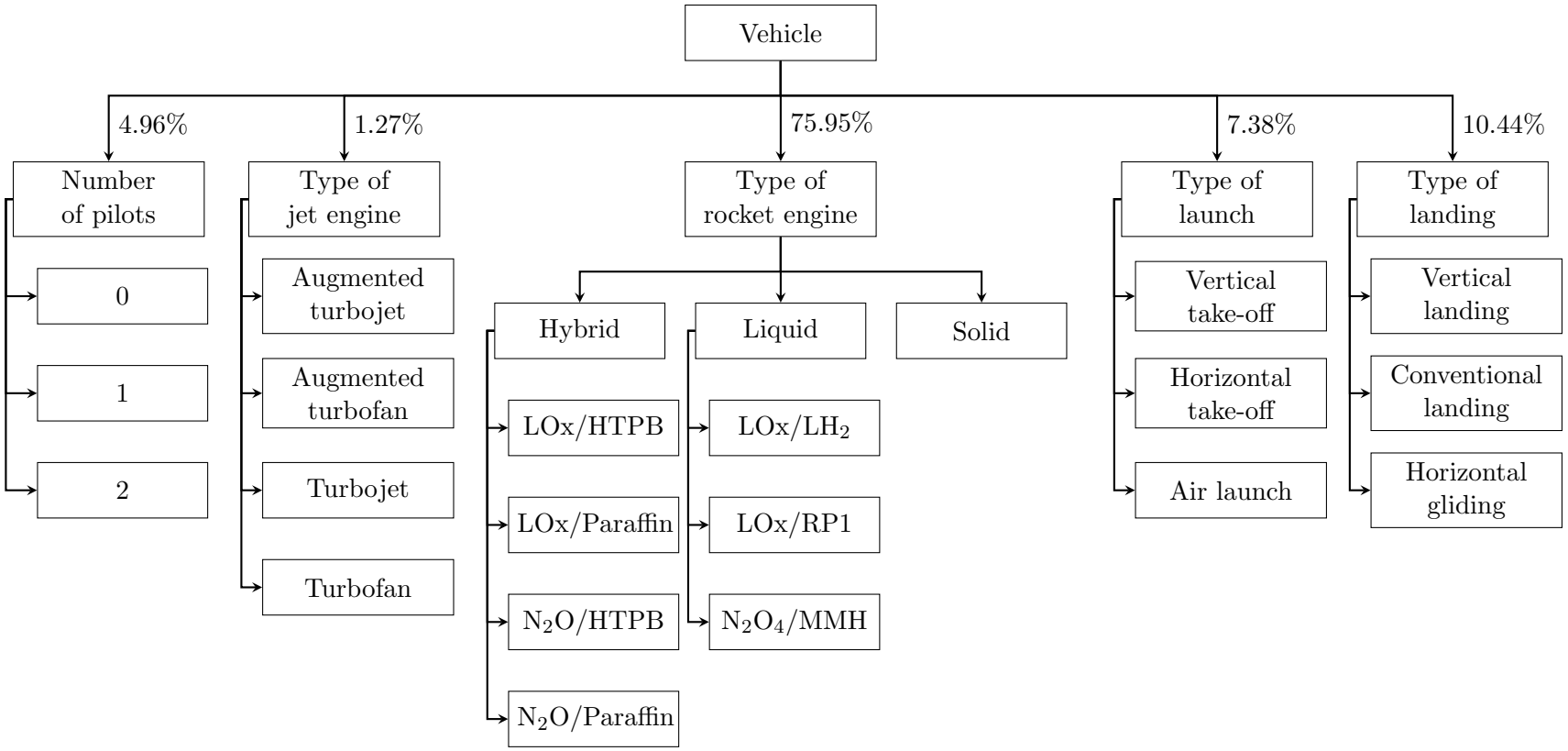



**Figure 84:** Safety analysis

## *6.8 Validation*

The design framework developed in this chapter has to be validated in order to be used for the overall optimization. This validation corresponds to Experiment 3. For that purpose, results obtained from the design framework are compared to actual data from existing vehicles. To fully validate the modeling and simulation environment, three vehicles are selected that cover a large portion of the design space:

- SpaceShipTwo from Virgin Galactic
- The New Shepard from Blue Origin
- The Rocketplane XP from Rocketplane Global Inc.

These vehicles are representative of the entire design space as the technologies used for these three vehicles cover most of the options identified in the morphological matrix, as shown in Table 42. The options chosen for each concept are identified by colors: SpaceShipTwo, New Shepard, and Rocketplane XP.

**Table 42:** Morphological matrix cover by the three existing vehicles

| | Option 1 | Option 2 | Option 3 | Option 4 | Option 5 | Option 6 | Option 7 | Option 8 |
|---|---|---|---|---|---|---|---|---|
| **Type of launch** | Horizontal | Vertical | Aircraft launched | Balloon launched | | | | |
| **Type of landing** | Horizontal powered | Gliding | Rocket | Parachute | | | | |
| **Lift generation** | Delta | Swept wing | Straight wing | None | | | | |
| **Longitudinal stability** | Horizontal stabilizer | Canards | None | | | | | |
| **Lateral stability** | Vertical stabilizer | Wing tip | None | | | | | |
| **Type of rocket engine** | Pressurized liquid | Pump-fed liquid | Solid | Hybrid | | | | |
| **Number of jet engines** | 0 | 1 | 2 | 3 | 4 | | | |
| **Type of jet engines** | Typical turbojet | Augmented turbojet | Typical turbofan | Augmented turbofan | None | | | |
| **Number of pilots** | 0 | 1 | 2 | | | | | |
| **Number of passengers** | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
| **Attitude control** | Cold gas | Liquid | | | | | | |

The validation process consists in inputting the geometric characteristics of the aforementioned vehicles along with their mission profile in the proposed design framework and comparing the results with actual data. The input data are displayed in Table 43.

The design framework outputs key parameters in terms of life-cycle costs and safety. However, due to the lack of information available in those fields, the take-off gross weight is used for the comparison, as it includes both the empty and fuel weights. The data used for the comparison are mostly taken from the vehicles' manufacturers databases [41, 362, 379]. The characteristics that have not been found are estimated using vehicles' 3-view drawings along with geometric and trigonometric considerations. For example, the length of both the front and the back parts have been estimated using the length of the entire vehicle. The comparison is based on relative errors on weight and length. The relative error on weight $\epsilon_{r,w}$ is calculated using Equation 145, where $W_{to,a}$ is actual take-off gross weight and $W_{to,c}$ the calculated take-off gross weight. The relative error on length $\epsilon_{r,l}$ is calculated using Equation 146, where $L_a$ is the actual length and $L_c$ the calculated length. The results of this comparison are presented in Tables 44 and 45.

$$\epsilon_{r,w} = \frac{|W_{to,c} - W_{to,a}|}{W_{to,a}} \tag{145}$$

$$\epsilon_{r,l} = \frac{|L_c - L_a|}{L_a} \tag{146}$$

For the three existing vehicles, the design framework provides very good results. Indeed, the mean relative error is equal to 11.2% with a maximum relative error of 22%. This accuracy is deemed acceptable for conceptual design level considerations. This validates the design framework.

Even though no data have been found to validate both the life-cycle costs and the risk level, the calculated values are analyzed to verify the consistency of the results. The latter are presented in Table 46. The program cost is calculated assuming that the three vehicles have been built and used once a month for 10 years.

**Table 43:** Inputs for the three existing suborbital vehicles [41, 252, 362, 379, 461]

| Input parameters | New Shepard | Rocketplane XP | Space-ShipTwo |
|---|---|---|---|
| **Maximum altitude (m)** | 132,000 | 104,000 | 100,000 |
| **Maximum load factor (g)** | 5.4 | 4 | 4 |
| **Space between two seats (m)** | 0.74 | 0.91 | 1.5 |
| **Number of passengers** | 4 | 5 | 6 |
| **Number of launches per year** | 12 | 12 | 12 |
| **Number of units built** | 3 | 3 | 3 |
| **Number of operating years** | 10 | 10 | 10 |
| **Wing** | No | Yes | Yes |
| **Take-off mode** | Vertical | Horizontal | Air launch |
| **Landing mode** | Vertical | Gliding | Gliding |
| **Transition altitude (m)** | - | 12,000 | - |
| **Number of pilots** | 2 | 1 | 2 |
| **Horizontal tail** | No | Yes | Yes |
| **Vertical tail** | No | Yes | Yes |
| **Jet engine** | No | Yes | No |
| **Chamber pressure (MPa)** | 3 | 4 | 3.3 |
| **Nozzle expansion ratio** | 50 | 20 | 25 |
| **Engine diameter (m)** | 6.98 | 1.7 | 2 |
| **Type of propellant** | LOx/$NH_2$ | LOx/RP1 | $N_2O$/HTPB |
| **Maximum thrust at sea level (N)** | 489,304 | 160,000 | 270,000 |
| **Number of jet engines** | - | 2 | - |
| **Maximum thrust per engine (N)** | - | 14,000 | - |
| **Bypass ratio** | - | - | - |
| **Afterburner** | No | Yes | No |
| **Turbine Inlet Temperature (K)** | - | 1,250 | - |
| **Fuselage diameter (m)** | 6.98 | 1.7 | 2 |
| **Fuselage base diameter (m)** | 1 | 1 | 1 |
| **Back part length (m)** | 0.5 | 0.5 | 0.5 |
| **Front part length (m)** | 2.22 | 0.5 | 0.5 |
| **Wing surface area ($m^2$)** | - | 30 | 35.82 |
| **Wing thickness-to-chord ratio** | - | 0.1 | 0.1 |
| **Sweep angle (rad)** | - | 0.78 | 0.96 |
| **Wing aspect ratio** | - | 6 | 1.88 |
| **Wing taper ratio** | - | 0.13 | 0.13 |
| **Vertical tail sweep angle (rad)** | - | 0.35 | 0.61 |
| **Vertical tail aspect ratio** | - | 3 | 3 |
| **Horizontal tail sweep angle (rad)** | - | 0.35 | 0.61 |
| **Horizontal tail aspect ratio** | - | 3 | 5 |

**Table 44:** Results of the weight validation of the design framework [41, 362, 379]

| | New Shepard | Rocketplane XP | SpaceShipTwo |
|---|---|---|---|
| **Calculated mass (kg)** | 74,842 | 11,100 | 14,903 |
| **Actual mass (kg)** | 65,043 | 9,072 | 13,154 |
| **Relative error (%)** | 13 | 22 | 13 |

**Table 45:** Results of the length validation of the design framework [41, 362, 379]

| | New Shepard | Rocketplane XP | SpaceShipTwo |
|---|---|---|---|
| **Calculated length (m)** | 24.2 | 13.27 | 17.7 |
| **Actual length (m)** | 21.1 | 13.41 | 18.3 |
| **Relative error (%)** | 14.7 | 1.0 | 3.3 |

**Table 46:** Estimated cost and risk level given by the design framework

| | New Shepard | Rocketplane XP | SpaceShipTwo |
|---|---|---|---|
| **Estimated program cost per passenger (Billions 2015 U.S. $)** | 25.5 | 1.91 | 0.92 |
| **Estimated risk level** | 462 | 458 | 394 |

Based on Table 46, the New Shepard appears to be the most expensive concept as it starts using its main rocket propulsion system from the ground. Moreover, its rocket engine needs to produce a very high thrust to move the new Shepard heavy structure, which results in expensive refuelings. In addition, it only carries four passengers, compared to five for the Rocketplane XP and six for the SpaceShipTwo, so the required take-off gross weight per passenger is larger, and consequently less efficient. The most economically viable solution is the SpaceShipTwo because it uses low-cost hybrid rocket engines and a carrier. In addition, the SpaceShipTwo also appears as the safest concept because it uses hybrid rocket engines. For comparison purposes, the risk level of the Space Shuttle has also been computed and is equal to 1,773. As expected, the various proposed concepts are safer than the Space Shuttle.

This section provides all the pieces necessary to evaluate the criteria identified for Experiment 3:

1. The calculated vehicles' performance is consistent with the literature, as the mean relative error on both weight and size is around 11%.

2. The execution time is acceptable, as it takes around 10 seconds to evaluate a single concept.

3. The modeling and simulation environment outputs flying, economic, and safety performance metrics as well as a complete geometric description of the vehicle. This provides a complete multi-disciplinary picture of the vehicle.

4. The modeling and simulation environment is capable of evaluating all types of suborbital vehicles.

These observations lead to the validation of Hypothesis 3:

**VALIDATION HYPOTHESIS 3: IF a sizing and synthesis environment based on a modified Multi-Disciplinary Feasible (MDF) approach and using both empirical relations and surrogate models is created THEN performance of the various alternatives of suborbital vehicles can be evaluated for further comparison and optimization.**

## *6.9 Identification of Key Trends*

This section aims at demonstrating the key capabilities of the proposed modeling and simulation environment by making new observations and providing key results about suborbital vehicles. As described in Section 6.3, the propulsion module bridges an important gap in the conceptual design of future chemical rocket engines. Hence, this module is used to provide insights in the performance, weight, and safety evaluation. As the proposed design framework is the first of its sort able to quantify the safety level of all suborbital vehicles, it is used to provide trends and insights in this domain.

### 6.9.1 Chemical Rocket Engine Design

One of the key contributions provided by the propulsion module is its ability to perform quantitative trade-offs among the various chemical rocket engines in terms of performance, empty weight, and safety. This section starts by providing important trends in the performance estimation of the three types of chemical rocket engines. Then, trade-offs that must be made between weight and performance are identified.

#### *6.9.1.1 Performance of Chemical Rocket Engines*

The use of surrogate models allows designers to identify new trends and perform both sensitivity and trade-off analyses among all types of rocket engines. It allows for the execution time to be reduced by a factor of $10^5$. This provides a great improvement when this performance estimation module is embedded in a large and complex optimization process. Figures 85 to 87 display the sensitivity of the performance parameters with respect to the key design variables for each type of propellant. As expected, liquid engines generally have better performance than solid and hybrid propellants. While the trends are similar for all liquid and hybrid propellants, one major difference can be noticed for solid propellants: $\epsilon$ has no impact on $c^*$ since $O/F$ does not vary. Moreover, the chamber pressure $p_c$ mainly drives $c^*$ and only has a small positive impact on Isp for all engine types. One should note that the variation of $O/F^*$ with $\epsilon$ is overestimated since a shifting equilibrium is assumed. With a frozen equilibrium during the expansion in the nozzle, the real curve of the variation of $O/F^*$ with $\epsilon$ has the same shape but with smaller values of $O/F^*$.

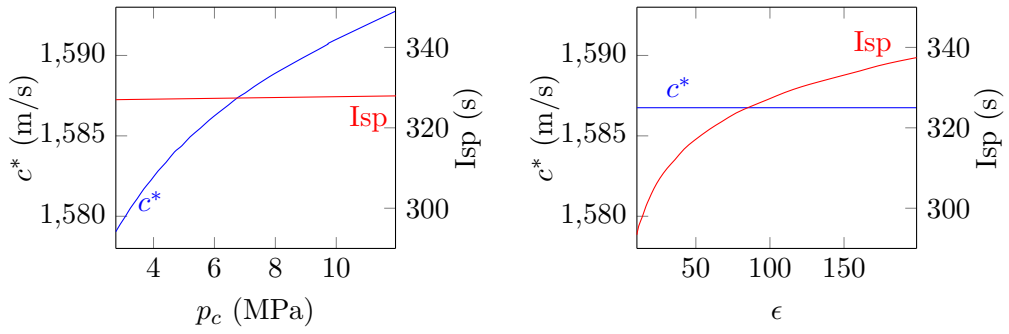



**Figure 85:** Sensitivity analysis of the performance of solid engines (Propellant C)

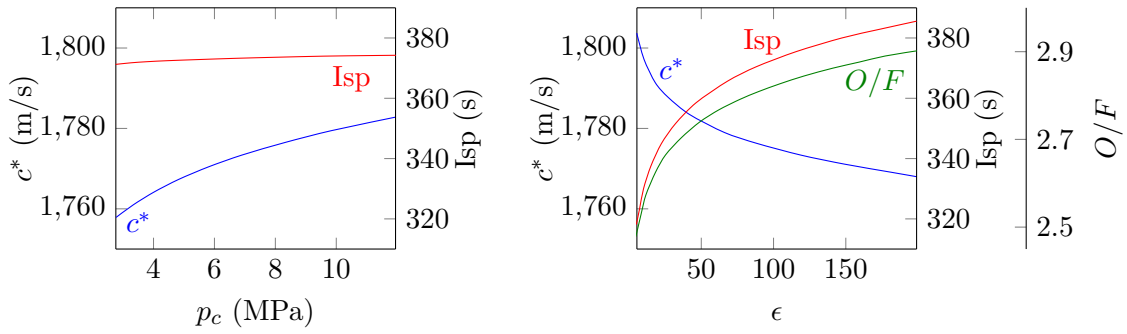



**Figure 86:** Sensitivity analysis of the performance of liquid engines ($O_2$/RP1)

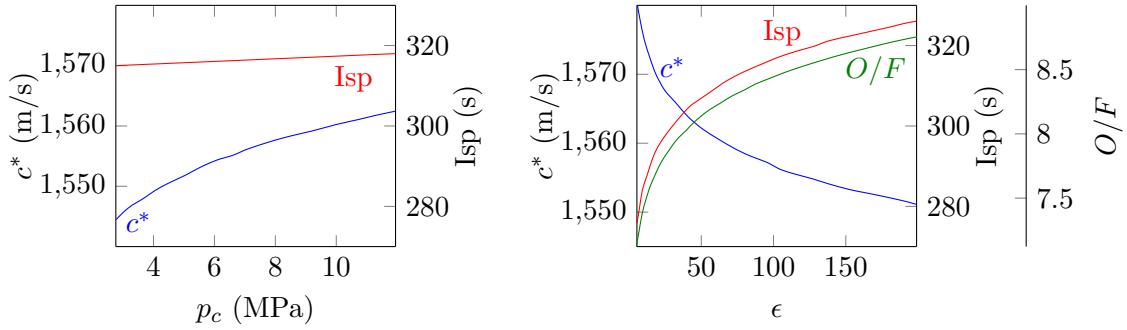



**Figure 87:** Sensitivity analysis of the performance of hybrid engines ($N_2O$/HTPB)

For solid propellant, the higher $p_c$ and $\epsilon$, the better the performance but the heavier and larger the engine. For liquid and hybrid engines, the trade-off is more complex. As $\epsilon$ and $p_c$ increase, the vacuum specific impulse $Isp_v$ improves. However, $c^*$ decreases when $\epsilon$ increases. Usually $c^*$ is independent of the nozzle conditions, but as $O/F^*$ increases with $\epsilon$ and $c^*$ decreases with $O/F^*$, $c^*$ decreases with $\epsilon$. However, the term $\epsilon \times c^*$ increases as epsilon increases because $c^*$ does not decrease much with $\epsilon$. Thus, the term $\epsilon \times c^*$ impacts the variations of the $Isp(p_a)$ with the ambient pressure $p_a$, or altitude via Eq. 147, where $p_a$ is the atmospheric pressure.

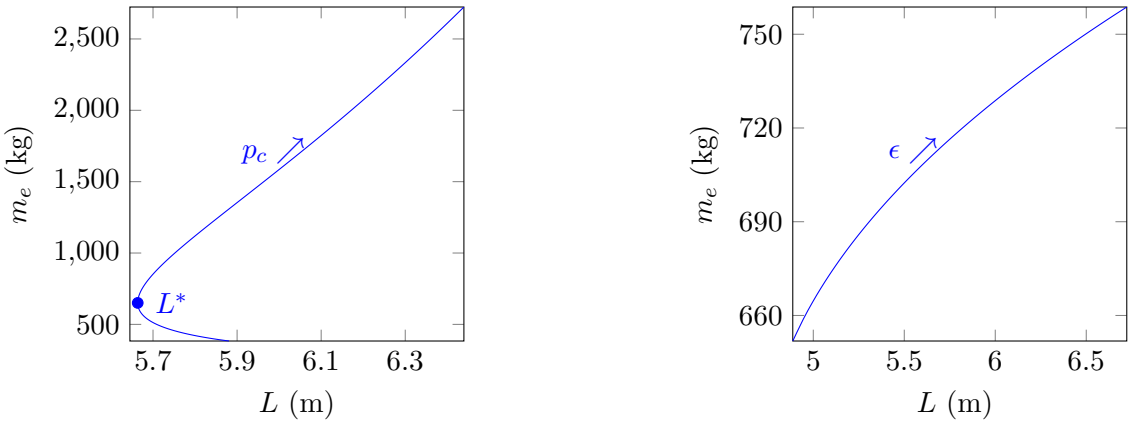
$$Isp(p_a) = Isp_v - \frac{\epsilon c^* p_a}{p_c g_0} \tag{147}$$

Consequently, a larger $\epsilon$ results in a less important improvement of $Isp(p_a)$ with altitude because of the increase of the second term in Eq. 147. Hence, there is a trade-off between high vacuum specific impulse $Isp_v$ and smaller improvement of $Isp(p_a)$ with altitude driven

by $\epsilon \times c^*$. In addition to these considerations, the impact of $p_c$ and $\epsilon$ on the engine weight must also be considered, as discussed in the next section.

#### *6.9.1.2 Trade-Offs Between Weight and Performance*

Using the approach described in Section 6.3, trade-offs between performance, weight, and length of rocket engines can be performed, as shown in Figure 88. In general, $p_c$ mainly impacts the weight, while $\epsilon$ drives the length. For $\epsilon = 55$, as $p_c$ increases from 2 to 12 MPa, even though the engine weight always increases, there is an optimum value $L^*$ for the engine length, which occurs for a small value of $p_c$ (4.57 MPa), as displayed in Figure 88(a). For $p_c = 5$ MPa, as $\epsilon$ increases from 5 to 200, both the engine weight and the engine length increase, so the lower the expansion ratio, the smaller and the lighter the engine (Figure 88(b)). This illustrates that the design of an optimized hybrid rocket engine is far from being trivial and requires that in-depth trade-off analyses be conducted between the weight and performance of the engine.




(a) $\epsilon = 55$, 2 MPa $\leq p_c \leq$ 12 MPa

(b) $p_c = 5$ MPa, $5 \leq \epsilon \leq 200$

**Figure 88:** Trade-offs between performance and weight for hybrid engines

#### *6.9.1.3 Application of the Propulsion Module*

When combined with the safety module, the propulsion module can support multi-disciplinary trade-offs among and across the different types of chemical rocket engines. To demonstrate this capability, the modules are used to find the best configuration of chemical rocket engine that fits into a fuselage with a two-meter diameter, while providing a 1,000-kN

thrust during two minutes. Three decision criteria are considered: engine weight, propellant cost, and risk. In order to identify potential trends, a DoE of 1,000 points has been generated following a Latin Hypercube model. The trade-offs that must be made between the different metrics are presented in Figures 89 and 90.

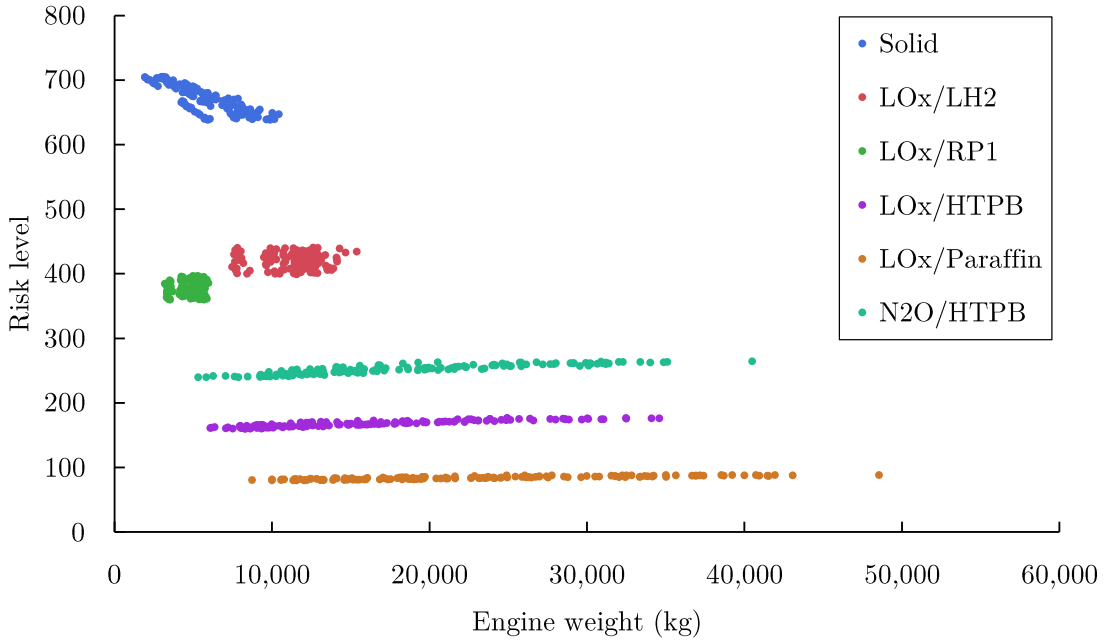



**Figure 89:** Propellant cost vs. engine weight

As expected, solid engines outscore both liquid and hybrid engines in terms of empty weight and propellant cost. However, since they are extremely risky and cannot be restarted, they are rarely used for manned vehicles. Comparing the other types of propellants, liquid engines tend to be lighter but more expensive in terms of propellant cost than hybrid engines. Figure 89 also shows the high sensitivity of hybrid engine weight and liquid engine operating costs with respect to changes in the chamber pressure and the nozzle expansion ratio.

Figure 90 demonstrates that, among hybrid engines, the mixture LOx/Paraffin is the safest but tends to require heavier engines. For liquid engines, Figures 89 and 90 show that the mixture LOx/RP1 is the safest and requires the lighter engines. However, this mixture is the most expensive. As a consequence, there are crucial trade-offs that must be made between the different objectives according to the design priorities.

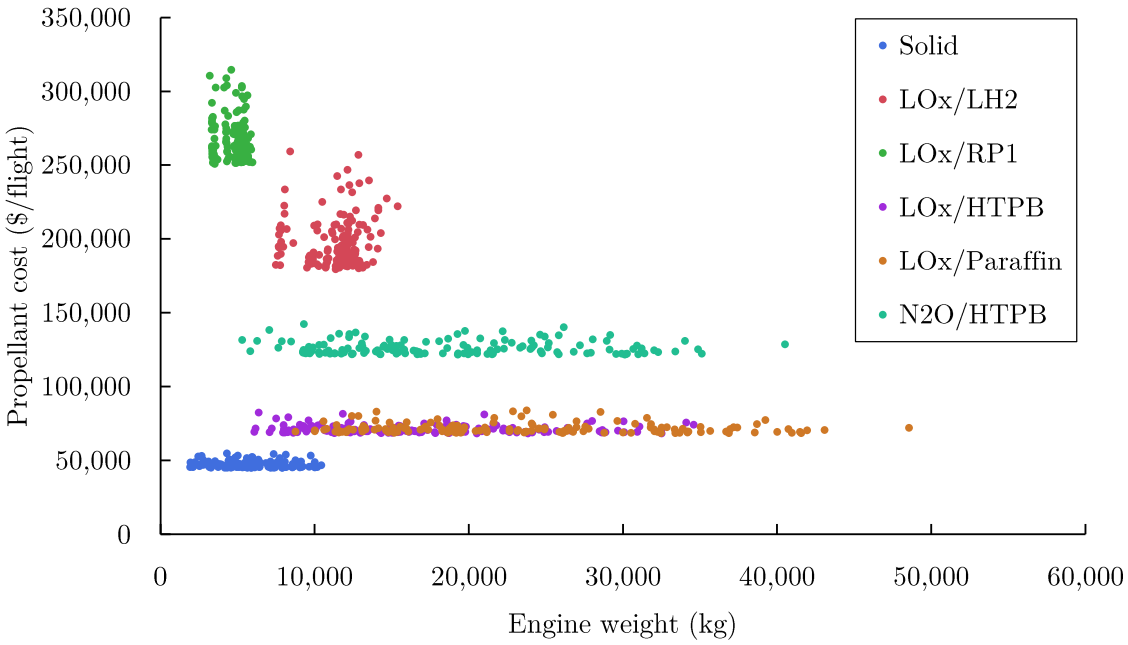



**Figure 90:** Risk level vs. engine weight

In order to find the best trade-off among these criteria, a probabilistic TOPSIS, as described in Section 7.3, is performed. An equal importance is given to each of the three aforementioned criteria, along with an uncertainty modeled with a standard deviation of 30%. The best engine appears to be a hybrid engine using LOx/Paraffin with a chamber pressure of 2.96 MPa and a nozzle expansion ratio of 71.

#### *6.9.1.4 Identification of the Key Drivers and Trends*

As described in the previous section, there are three main design variables when developing a new rocket engine: the mixture of propellants, the nozzle expansion ratio, and the chamber pressure. The relative impact of each of these variables on the three key metrics is presented in Figure 91. As demonstrated by the presence of clusters represented by the different colors, the major design driver is the propellant mixture.

In addition to the mixture selected, the key weight and safety driver appears to be the combustion chamber pressure, while the nozzle expansion ratio mainly drives the propellant cost. Trade-offs between the different propellant mixtures can also clearly be identified. For

example, the three engines that provide the lowest engine weight (solid propellant, LOx/-Paraffin, and $N_2O$/HTPB) are the riskiest.

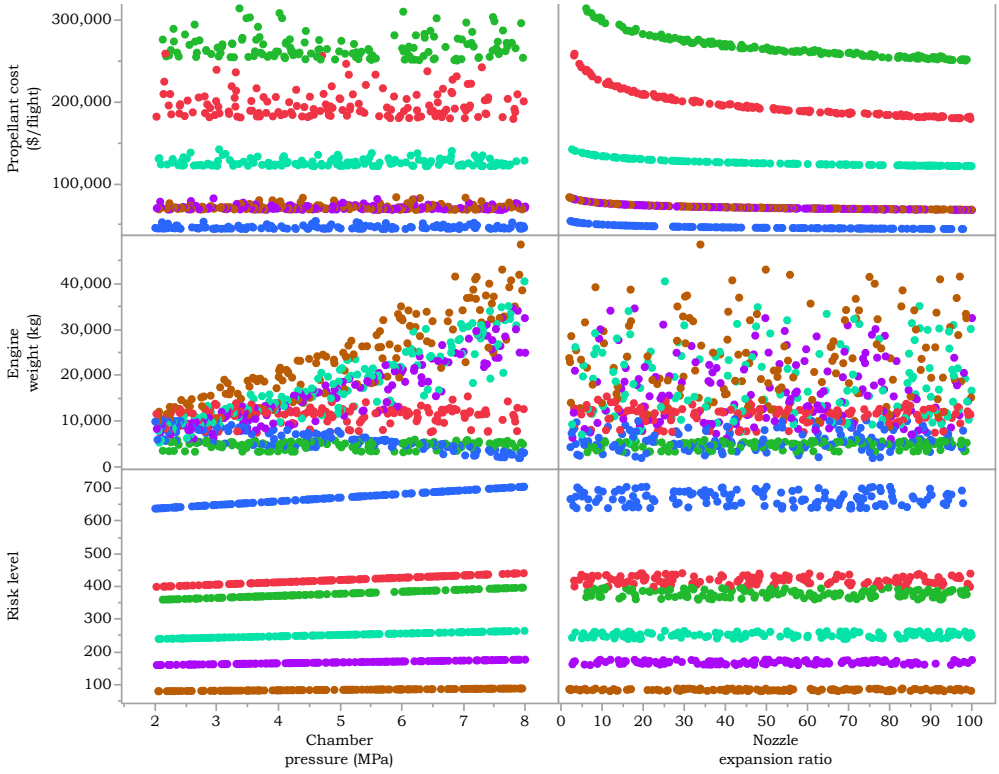



**Figure 91:** Sensitivity analysis of the propulsion system parameters

### 6.9.2 Safety Analysis

The safety module developed in Section 6.7 is used to identify the key safety drivers and perform new observations. In addition, those results will be used to check the consistency and the correct behavior of the safety module.

First, a Latin Hypercube DoE is developed to generate points that cover the entire design space. Based on this sampling of the design space, the key safety drivers are identified through a screening with JMP. The main safety drivers are presented in Figure 92. As shown, the main safety drivers are the type of propellant *Propellant*, the type of landing *LAmode*, the type of take-off *TOmode*, the number of pilots $n_{Pilots}$, the rocket engine

chamber pressure $p_c$, and the rocket engine thrust $T_r$. As expected, the type of propellant is, from far, the main safety driver. Indeed, it is the most probable cause of catastrophic failure in a spacecraft.

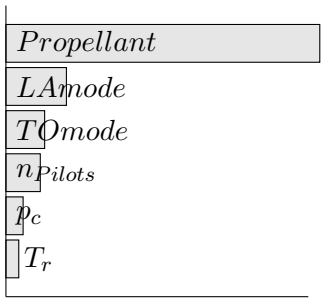



**Figure 92:** Relative importance of the design variables on the risk level

Figure 93 displays the risk level as a function of the type of propellant and for the different take-off modes. As expected, three clusters emerge, which can be directly linked to three main levels of risk:

- High: solid propellants with a risk level above 650
- Medium: liquid propellants with a risk level between 400 and 650
- Low: hybrid propellants with a risk level below 400

Moreover, for each propellant, the type of take-off highly impacts the risk level: launched from an aircraft appears to be the riskiest take-off mode, while the vertical is the safest.

Figure 93 clearly shows that a vehicle powered by a solid engine would not be able to reach a good safety level, whatever the other design choices. It also shows the sensitivity of the mixture on safety within each propellant category: liquid and hybrid. In particular, hypergolic mixtures are the riskiest among all liquid mixtures, while the mixture $N_2O$/HTPB is the riskiest among all hybrid mixtures. As displayed, the type of mixture within each propellant category has approximately the same impact as all other variables put together. Indeed, the maximum difference between the mean safety level of each mixture is close to the range of safety levels of each mixture.

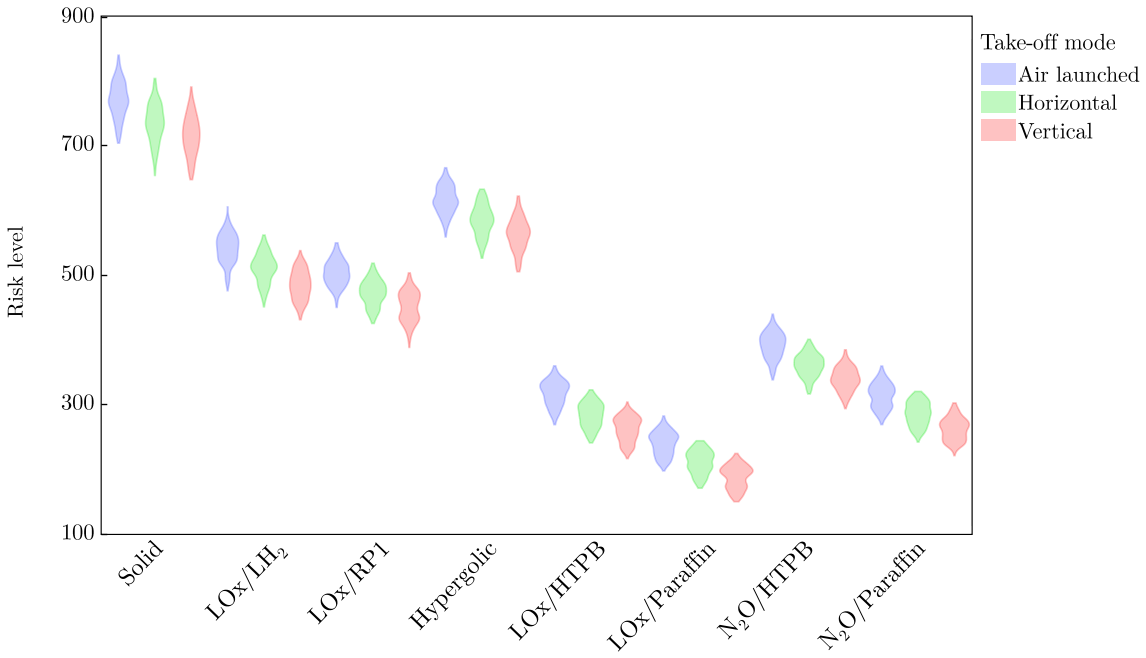



**Figure 93:** Risk level vs. type of propellant

Finally, once a mixture of propellants has been selected by the designers, the uncertainty on the safety level remains relatively low. Table 47 provides the mean value, the standard deviation, and the range for the distribution of risk level for each mixture.

**Table 47:** Risk level distributions

| **Propellant** | **Mean risk level** | **Standard deviation** | **Range** |
|---|---|---|---|
| Solid | 742 | 37.5 | 650 - 839 |
| LOx/$LH_2$ | 512 | 32.6 | 432 - 606 |
| LOx/RP1 | 475 | 30.0 | 390 - 550 |
| Hypergolic | 590 | 31.6 | 507 - 666 |
| LOx/HTPB | 285 | 29.8 | 217 - 359 |
| LOx/Paraffin | 211 | 27.9 | 150 - 281 |
| $N_2O$/HTPB | 364 | 28.6 | 294 - 439 |
| $N_2O$/Paraffin | 286 | 27.8 | 221 - 358 |

This chapter discussed the development of the individual modules that compose the design framework. They have been embedded into a single sizing and synthesis environment, which will be used by a decision-making process, as described in the next chapter.

# CHAPTER VII

# STEP 4: MAKE INFORMED DECISIONS

This chapter discusses the development of a decision-making environment capable of helping designers with the different phases of the design process. Indeed, as mentioned in Section 3.4, this environment must facilitate the decision process by providing decision makers with information regarding the performance, economic viability, safety, and robustness of the different alternatives of interest. While this environment has to support trade-off analyses and rapid prioritization, it should also allow decision makers to have a rapid overview of the selected architectures. As a consequence, three capabilities must be integrated into the decision-making environment:

1. Trade-off capabilities: enable the visualization of Pareto frontiers and multidimensional data to highlight trades between the multiple objectives. In particular, such visuals are critical to the identification of dominant architectures or technologies in the various regions of the solution space.

2. Ranking and prioritization capabilities: a ranking is obtained by fixing the relative importance of each objective and evaluating each alternative against the formulated objective function.

3. A parametric visualization CAD model: to provide decision makers with a three-dimensional rendering of the concepts considered.

Figure 94 describes the overall architecture of the decision-making framework, where the specific capabilities that will be used by decision makers are displayed in blue and the modules that must be developed in yellow. The development of these aforementioned modules is discussed in the next sections, followed by the application of the proposed decision-making environment to the design of affordable and safe suborbital vehicles.

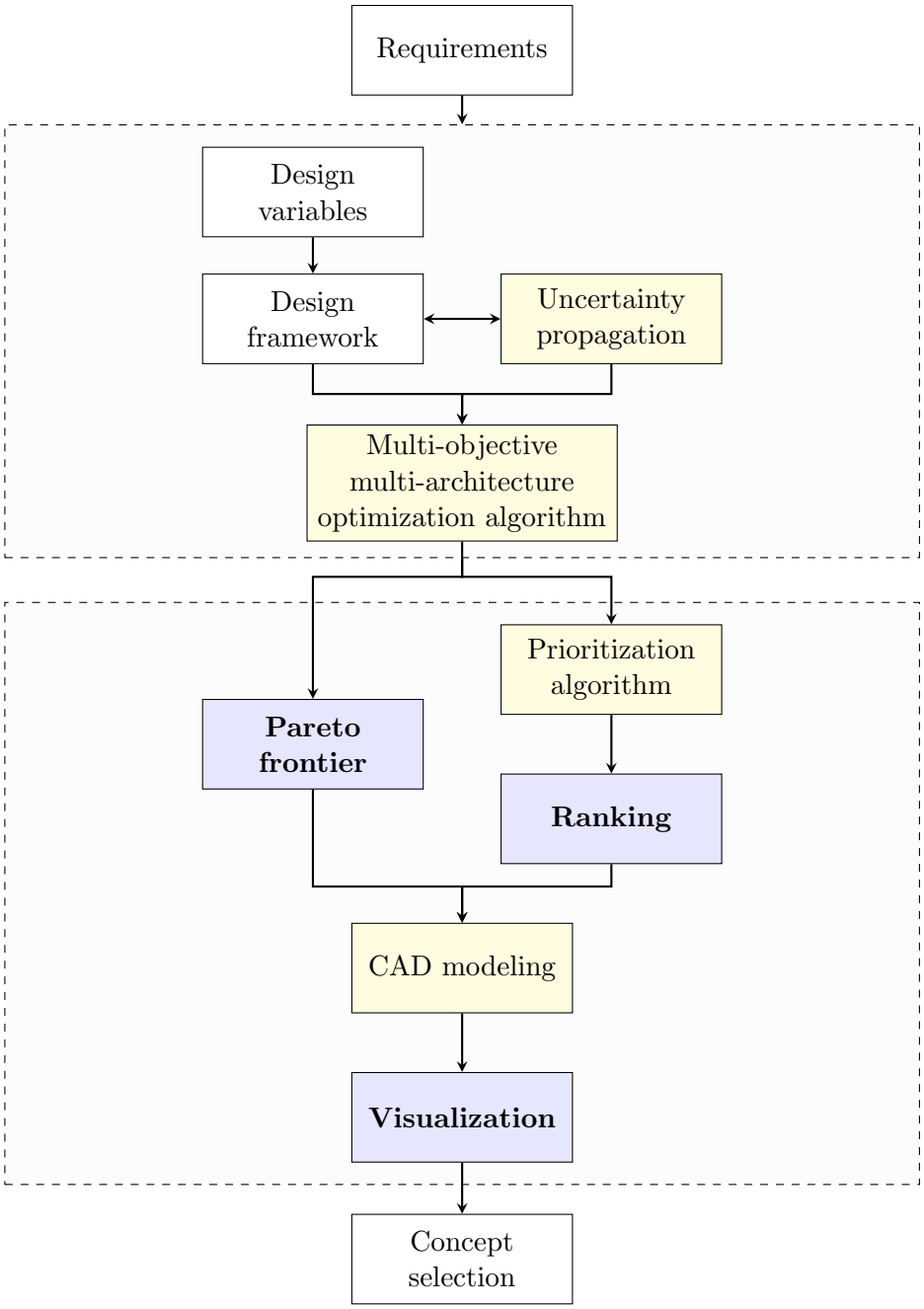



**Figure 94:** Architecture of the decision-making environment

### *7.1 Uncertainty Propagation*

As discussed in Section 2.2, when dealing with emerging markets, a large amount of uncertainty is present in the requirements. Hence, there is a need to include robustness in the decision-making process. For that purpose, fuzzy set theory is used to model and

propagate uncertainty throughout the design framework. The approach used to model the degree of uncertainty in requirements has already been discussed in Section 4.2. This section aims at discussing the approach chosen to propagate uncertainty.

For a problem with $n$ uncertain design variables $x_i$, each variable is characterized by a mean value $\mu_i$ and a standard deviation $\sigma_i$. If $X$ is defined as the objective function such that $X = f(x_1, x_2, ...x_n)$, then the variance of X, Var$(X)$, can be defined using Equation 148, where $g_i = \frac{\partial f}{\partial x_i}$ is the partial derivative of $f$ with respect to $x_i$ and $\sigma_{ij}$ is the covariance of $x_i$ and $x_j$.

$$\text{Var}(X) = \sum_{i=1}^{n} \sum_{j=1}^{n} g_i(\mu) g_j(\mu) \sigma_{ij} \tag{148}$$

In addition, the covariance $\sigma_{ij}$, which is a measure of how much two variables change together, can be calculated using Equation 149. In particular, the covariance of $x_i$ and $x_j$ is 0 if $x_i$ and $x_j$ are independent.

$$\sigma_{ij} = \begin{cases} \text{cov}(x_i, x_j) & \text{if } i \neq j \\ \sigma_i^2 & \text{if } i = j \end{cases} \tag{149}$$

Since there is no available analytical solution for $g_i$, a numerical approach is necessary. For that purpose, the Newton's difference quotient, also known as the first-order divided difference, is used, as illustrated in Equation 150. In this equation, $h$ is a small number fixed at $10^{-3}$ and $h_i$ is a vector with zero for each component except for the i$^{\text{th}}$ component, which is equal to $h$.

$$g_i(\mu) = \frac{\partial f(\mu)}{\partial x_i} \approx \frac{f(\mu + h_i) - f(\mu)}{h} \tag{150}$$

This uncertainty model has to be called by the multi-objective optimization algorithm, when evaluating the objective function. The development of this multi-objective optimization framework is further discussed in the next section.

## *7.2 Evolutionary Multi-objective Multi-architecture Algorithm (EMMA)*

The goal of this section is to develop an approach that allows designers to efficiently optimize concepts that are not described by the same design variables against multiple and

competing objectives. As discussed in Section 2.1.3.4, the proposed algorithm called EMMA can be described using the following four-step process:

1. Individually optimize each architecture using specifically configured NSGA-II with an initial number of generations common to all architectures.

2. Evaluate the performance of each architecture compared to others based on their location on the overall Pareto frontier.

3. Re-execute each NSGA-II with a specific number of generations given by an evolutionary algorithm as a function of the performance of its corresponding architecture.

4. Repeat steps 2 and 3 until the convergence criterion is met.

The overall optimization process is summarized in Figure 95. The following sections describe both the NSGA-II and the new evolutionary algorithm, as well as their integration and validation.

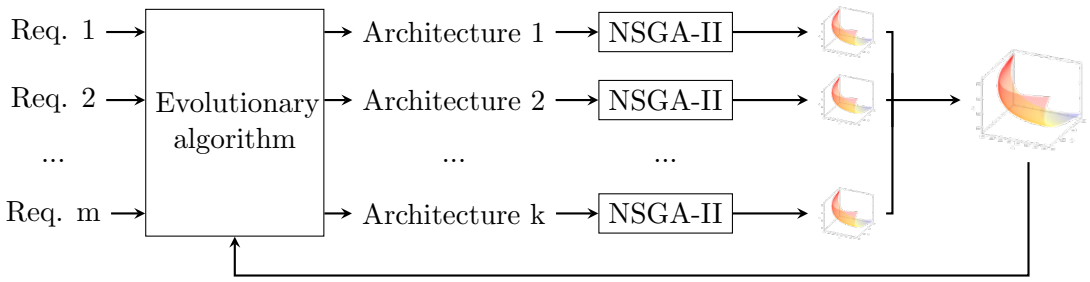



**Figure 95:** Multi-objective multi-architecture optimization process

### 7.2.1 Non-Dominated Sorting Genetic Algorithm-II

As discussed in Section 3.4, the NSGA-II is chosen to optimize each architecture. First, it is able to handle both discrete and continuous variables, and can deal with complex and thus potentially non-convex design spaces. Moreover, it is independent of predefined settings, which is a great advantage since the behavior of the response is completely unknown at this point. The NSGA-II enables an evenly-spaced sampling of the Pareto frontier and is very efficient when dealing with unconventionally-shaped solution spaces. Even though

this approach is relatively resource and time consuming, it has the capability of handling all kinds of design spaces originating from complex problems. The general procedure of the NSGA-II is described by Deb in Figure 96 [110].

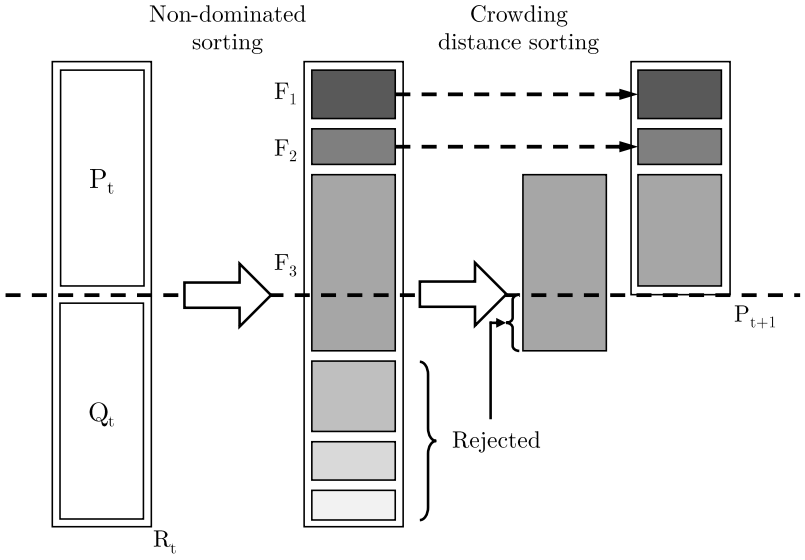



**Figure 96:** General NSGA-II process [110]

The NSGA-II starts with a population $P_t$ of $n$ individuals randomly generated within the design space. Then, using the same operating principles as the genetic algorithm, crossovers and mutations are performed on the population $P_t$ in order to generate a new population $Q_t$ composed of $n$ new individuals. The population $R_t$, composed of $2n$ individuals, is then created by combining both the populations $P_t$ and $Q_t$. Then, the selection of the individuals that will be part of the new population is based on their fitness. The latter depends on two measures: the non-domination level (measure of relative dominance) and the crowding distance (measure of relative isolation). These two measures are performed following two steps:

- Non-dominated sorting: individuals are grouped and ranked based on their domination level ($F_1$, $F_2$, etc.).
- Crowding distance sorting: individuals are ranked within each domination level $F_i$.

Following these two steps, only the top $n$ individuals are kept and compose the new population $P_{t+1}$. The next sections discuss the two sorting steps.

#### *7.2.1.1 Non-Dominated Sorting*

For the non-domination sorting step, the fitness (or performance) of each individual is evaluated based on its domination level (1 is the best level, 2 the next best level, etc.). To do so, an algorithm is first executed on all points to find all non-dominated solutions. A fitness of 1 is assigned to those points and they are "set aside" from the population. Then, the algorithm is executed again to find the non-dominated points and a fitness of 2 is assigned to the non-dominated points. This process is executed until a fitness value has been assigned to all individuals. As a consequence, all individuals are grouped and ranked based on their domination level. Once grouped into non-domination level categories, individuals need to be ranked within each category, as described in the following section.

#### *7.2.1.2 Crowding Distance Sorting*

The crowding distance sorting aims at measuring how isolated each point of the population is. The crowding distance of a given point corresponds to the average distance of its two neighboring points, as shown in Figure 97.

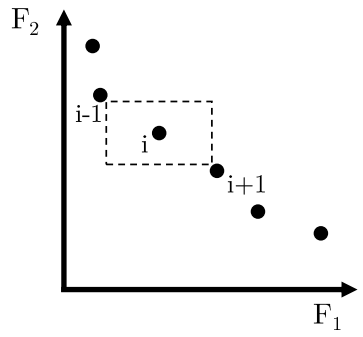



**Figure 97:** Crowding distance calculation [110]

By preferring isolated points (the ones with the largest crowding distance), the final points on the Pareto frontier tend to be more evenly spaced. Hence, individuals are ranked based on their crowding distance within each domination level from the largest to the smallest.

This two-step process is repeated until the convergence criterion is reached. This results

in providing the best set of configurations for each architecture with respect to several objectives. In order to drive this process across the different architectures, an evolutionary algorithm is developed, as discussed in the next section.

### 7.2.2 Evolutionary Algorithm

While the NSGA-II can be individually applied to each architecture, there is a need to combine all architectures. For that purpose, a new evolutionary algorithm, inspired by the Simulated Annealing [53, 54], is developed that drives multiple NSGA-IIs based on their performance. The objective is to favor promising architectures without neglecting others. The new algorithm is defined by a four-step process:

1. Execute the NSGA-II for each architecture with an initial number of generations $N_0$ similar for all architectures.

2. For each architecture $i$, evaluate the fitness $f_i$ equal to the number of non-dominated points originating from architecture $i$ in the overall Pareto frontier.

3. Determine the number of generations $N_i$ of each architecture for the next iteration of the primary algorithm based on the following principle:

   - If $f_i \neq 0$: $N_i$ is proportional to $f_i$, as described in Equation 151.

   $$N_i = \left\lfloor \frac{f_i}{\sum_i f_i} \right\rfloor N_0 \tag{151}$$

   - If $f_i = 0$: $N_i$ has a probability $p$ to reach $N_0$. The parameter $p$ is defined in Equation 152, where $m$ is the number of fruitless iterations and $T$ a given constant.

   $$p = 1 - \exp\left(\frac{-m}{T}\right) \tag{152}$$

4. Repeat steps 2 and 3 until the convergence criterion is met.

### 7.2.3 Integration

In order to link these two optimization algorithms, a dedicated optimization framework is developed, as presented in Figure 98. The general optimization parameters along with

the number of architectures $N$, the population size, the number of design objectives, and the number of iterations are integrated into the main optimization function.

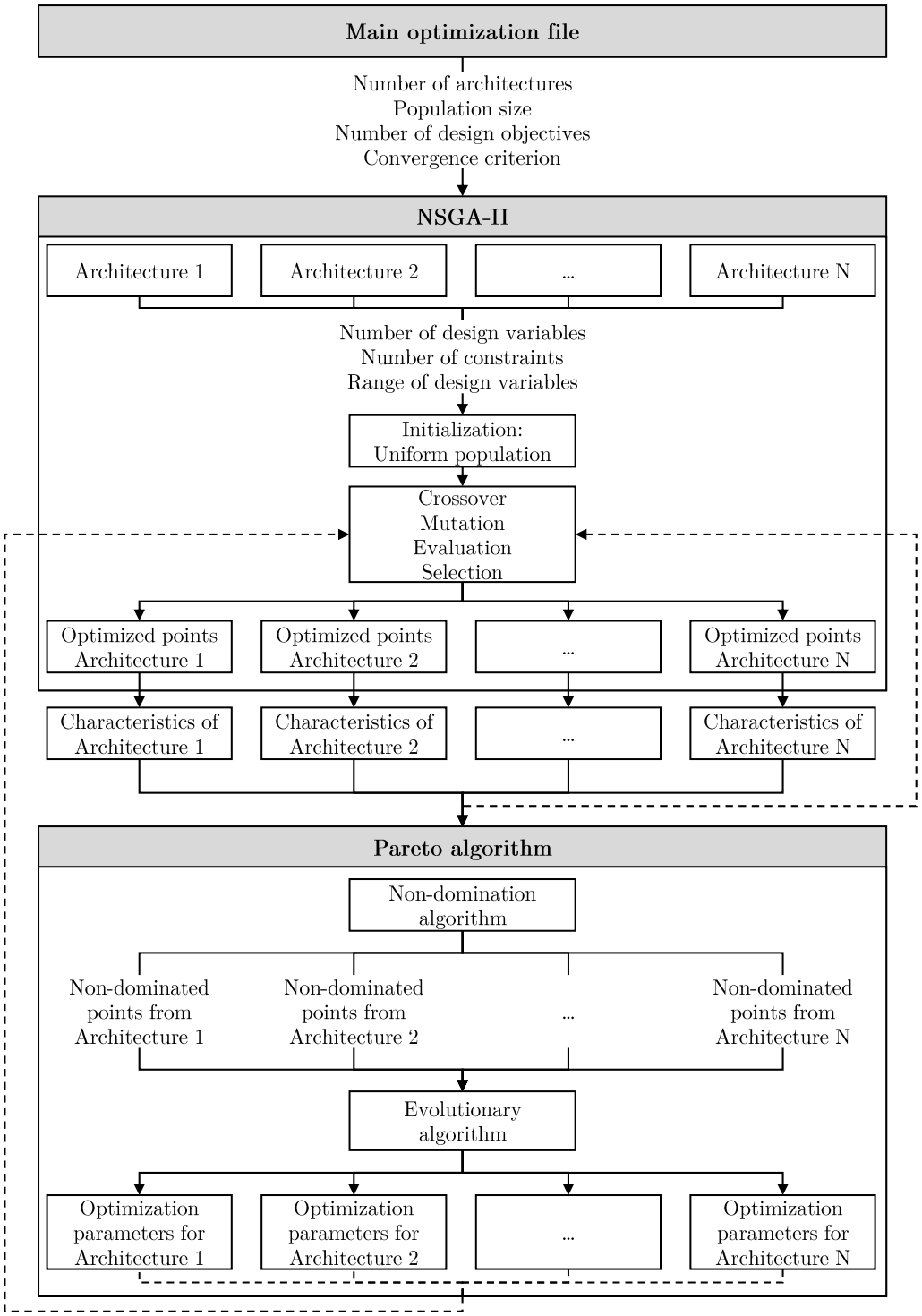



**Figure 98:** Architecture of the overall multi-disciplinary optimization framework

For consistency purposes, the population size is fixed at a given value for all architectures. The main optimization function then drives the multiple NSGA-IIs via $N$ architecture files. Those files detail the distinctive characteristics of each architecture such as the number of constraints as well as the number of design variables along with their ranges. Then, the NSGA-II is executed for each architecture using the design framework developed in the previous chapter. While the design framework might be specific to certain architectures, the structure of the optimization algorithm is common to all architectures. This facilitates the interchangeability and the potential updates of the optimization platform. Each of the NSGA-II algorithms outputs a file with a detailed description of all points on the Pareto frontier.

These files are combined to generate the new population, which is inputted into a non-domination algorithm. This algorithm extracts the non-dominated points, while keeping track of their original architecture. This algorithm provides the evolutionary algorithm with the fitness of every architecture. Then, the evolutionary algorithm determines the number of generations assigned to each NSGA-II for the next iteration. Finally, based on both the number of generations and the initial points, the multi-objective optimization algorithm is executed until the convergence criterion, which is fixed by designers, is met.

This optimization platform has been developed in Matlab, as described in Appendix D.3. In order to speed up the process, the overall algorithm has been parallelized on multiple computers. Indeed, each architecture is optimized by a given computer in parallel. In addition, the optimization of each architecture has been parallelized using the different cores of each computer. Hence, the objective functions are evaluated in parallel. The next section applies the proposed process on multiple test functions in order to both validate and quantify the benefits of the proposed algorithm.

### 7.2.4 Validation and Results

For validation purposes, Experiment 1.2 is set using the proposed algorithm to find the Pareto frontier of well-known test functions. To account for the fact that multiple architectures must be simultaneously optimized, each test function is duplicated and slightly

modified. In order to assess the benefits of the new approach, the results are compared to the Pareto frontier that would be provided using a DoE with the same number of function calls. In order to benefit from a rich sampling of the interior of the design space, a Latin Hypercube DoE is chosen. The objective is to show that the proposed algorithm provides a better and more accurate sampling of the Pareto frontier than the DoE. To quantify the benefits, the Pareto frontiers determined with both approaches are combined and only the non-dominated points are kept. Two evaluation criteria are found relevant for this comparison: accuracy and quality of the sampling. First, the accuracy is measured by comparing the number of points on the Pareto frontier originating from the proposed algorithm with the number of points originating from the DoE. This comparison is quantified using the ratio $\eta_o$, which represents the percentage of non-dominated points coming from the proposed algorithm over the number of points coming from the DoE. Then, the quality of the sampling is also evaluated by looking at the distribution of points on the Pareto frontier. The goal is to show that the points generated by the proposed algorithm are more evenly sampled than the ones generated with the DoE. The following sections discuss the results obtained for two different test functions.

#### *7.2.4.1 Two-Dimensional Problem with a Large Solution Space*

The efficiency of the proposed algorithm is tested on a two-dimensional function with a relatively large design space. For that purpose, Poloni's two objective function [91] is selected as the baseline, as presented in Equation 153, where $a_1$, $b_1$, $a_2$, $b_2$, $c_2$, $d_2$, $a_3$, $b_3$, $c_3$, and $d_3$ are ten constants.

$$\begin{aligned} &\min \begin{cases} f_1(x,y) = 1 + (a_1 - g(x,y))^2 + (b_1 - h(x,y))^2 \\ f_2(x,y) = (x+3)^2 + (y+1)^2 \end{cases} \\ &\text{where} \begin{cases} g(x,y) = a_2 \sin(x) - b_2 \cos(x) + c_2 \sin(y) - d_2 \cos(y) \\ h(x,y) = a_3 \sin(x) - b_3 \cos(x) + c_3 \sin(y) - d_3 \cos(y) \\ -\pi \leq x, y \leq \pi \end{cases} \end{aligned} \tag{153}$$

Four different architectures are created by changing the set of variables ($a_1$, $b_1$, $a_2$, $b_2$, $c_2$, $d_2$, $a_3$, $b_3$, $c_3$, $d_3$). Figure 99 shows the results of both the DoE and the proposed optimization algorithm as well as the optimal Pareto frontier.

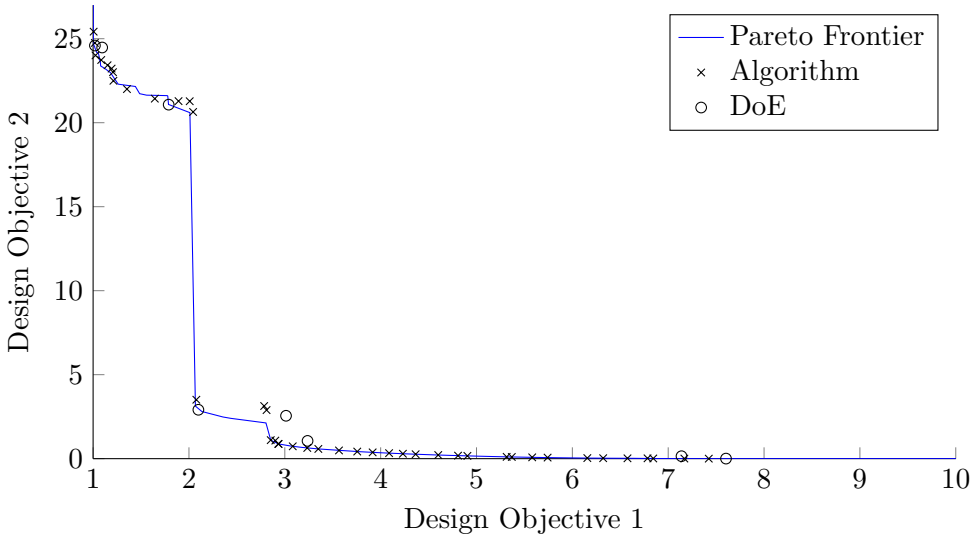



**Figure 99:** Validation of the algorithm for a two-dimensional problem

As expected, the proposed algorithm results in a better and more evenly-spaced sampling of the Pareto frontier. The calculation of the benefits leads to $\eta_o = 93\%$.

#### *7.2.4.2 Three-Dimensional Problem with a Large Design Space*

In order to test the algorithm on a three-dimensional problem, a set of functions with three objectives is created based on the Viennet function [91]. The design objectives are defined as a function of two design variables $x$ and $y$, as described in Equation 154, where $a_1$, $b_1$, $a_2$, $b_2$, $a_3$, and $b_3$ are six constants.

$$\min \begin{cases} f_1(x,y) = a_1\left(x^2+y^2\right) + b_1 \sin\left(x^2+y^2\right) \\ f_2(x,y) = a_2\left(3x-2y+4\right)^2 + b_2\left(x-y+1\right)^2 + 15 \\ f_3(x,y) = \frac{1}{x^2+y^2+a_3} - b_3 \exp\left(-\left(x^2+y^2\right)\right) \end{cases} \quad \text{where } -3 \leq x,y \leq 3 \tag{154}$$

Four different architectures are created by changing the different variables from the following set ($a_1$, $b_1$, $a_2$, $b_2$, $a_3$, $b_3$). Figure 100 shows the results of both the DoE and the proposed optimization algorithm as well as the optimal Pareto frontier.

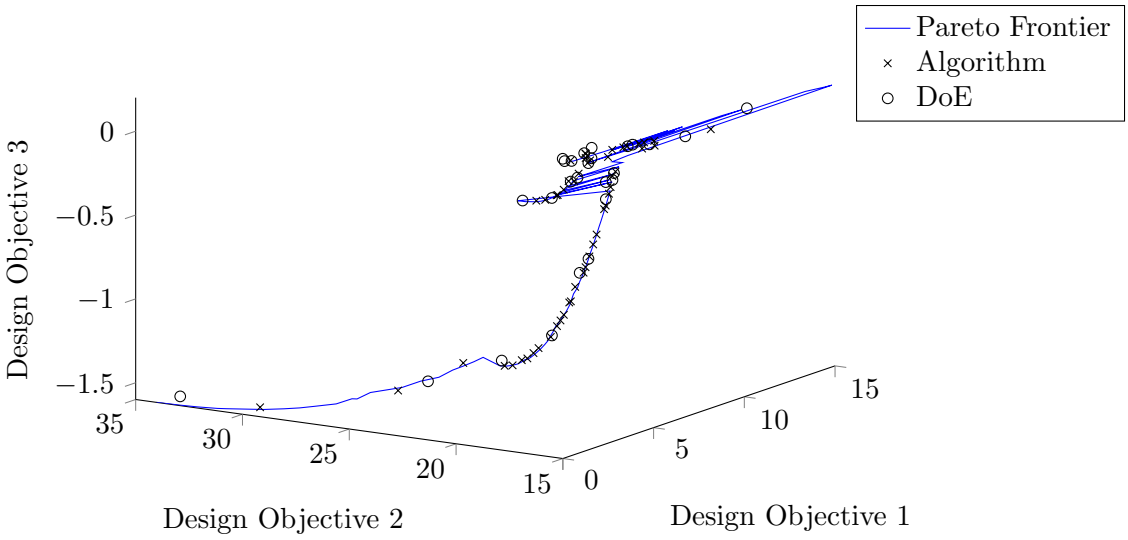



**Figure 100:** Validation of the algorithm for a three-dimensional problem

As expected, the proposed algorithm also results in a better and more evenly-spaced sampling of the Pareto frontier. The calculation of the benefits leads to $\eta_o = 67\%$.

### 7.2.5 Quantification of the Benefits

The proposed algorithm might be complicated to implement and might have some limitations. Indeed, for small and low-complexity problems, the benefits of the proposed algorithm might not be significant. Hence, this section aims at providing a quantitative measure of the complexity of the problem $C_p$ so that designers have an easy way to evaluate the usefulness of the proposed algorithm for their specific problems. The development of this complexity factor is based on two main observations:

- The larger the design space, the smaller the probability for the DoE to select a combination of design variables on the Pareto frontier.
- The larger the solution space, the smaller the probability for the DoE to produce an accurate and evenly-spaced Pareto frontier.

Based on these two observations, the complexity factor mainly depends on the size of both the solution and the design spaces. As a consequence, the general form of the complexity factor is presented in Equation 155, where $C_{p,d}$ is the design space complexity factor and

$C_{p,s}$ the solution space complexity factor.

$$C_p = C_{p,d} \times C_{p,s} \tag{155}$$

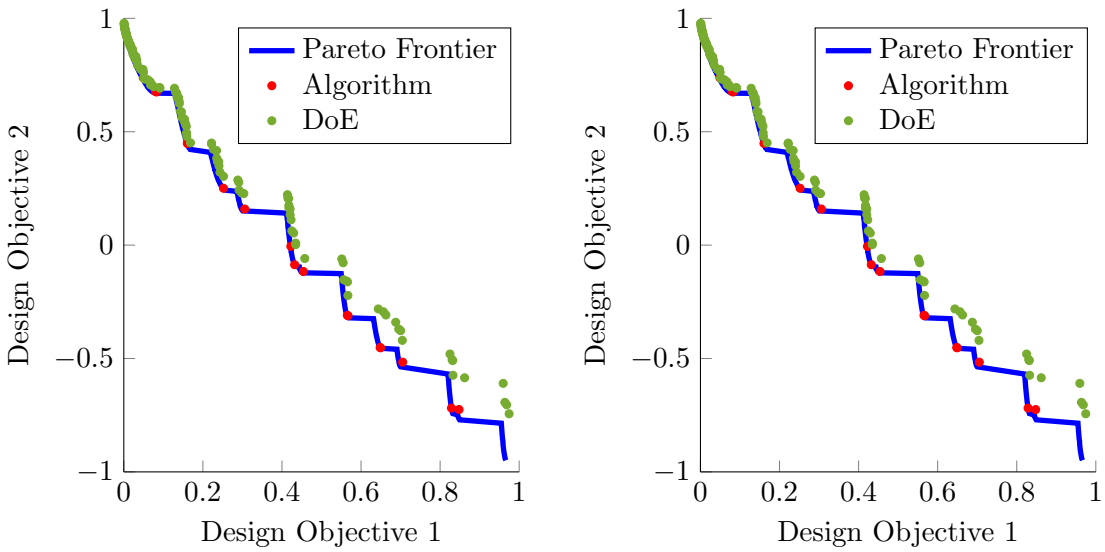
The size of the normalized design space is proportional to its number of dimensions, so that $C_{p,d}$ is assumed to be proportional to the number of dimensions $k$ of the design space. In addition, in order to obtain an order of magnitude of the size of the solution space, both the number of dimensions and the size of each dimension must be taken into account. The size of each dimension can be evaluated using the range of possible solutions defined by $f_{i,max} - f_{i,min}$. If the range of solutions on one dimension is close to zero, this dimension should neither decrease nor increase the size of the solution space. As such, 1 is added to the previous difference. As a consequence, $C_{p,s}$ is modeled using Equation 156, where $f_{i,max}$ and $f_{i,min}$ are the maximum and minimum of the i$^{\text{th}}$ objective, respectively.

$$C_{p,s} = \prod_{i=1}^{n} (f_{i,max} - f_{i,min} + 1) \tag{156}$$

Thus, combining the models for both the solution space and the design space complexity factors, $C_p$ can be described using Equation 157.

$$C_p = k \times \prod_{i=1}^{n} (f_{i,max} - f_{i,min} + 1) \tag{157}$$

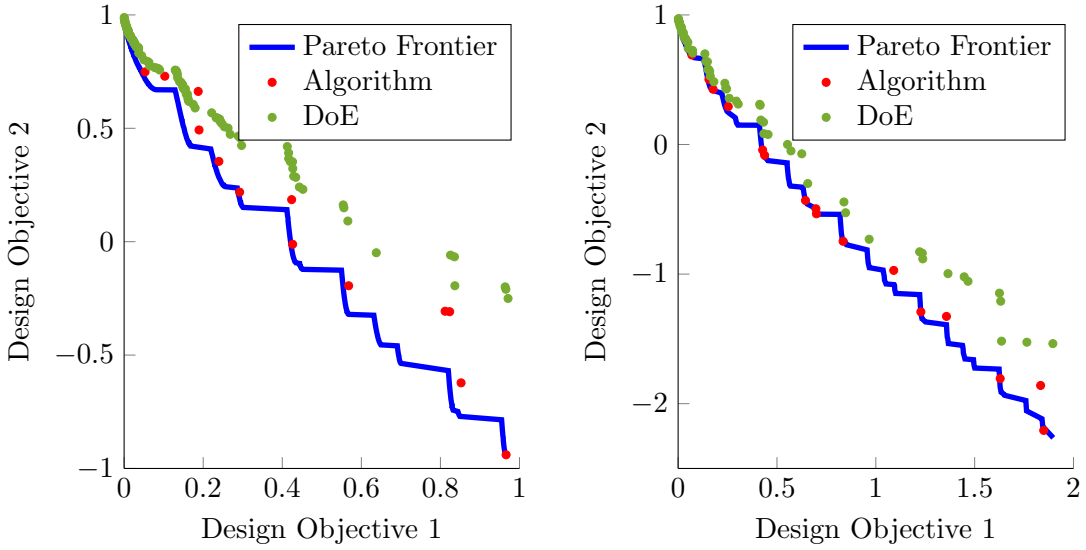
In order to assess the accuracy of the proposed criterion, a series of tests is performed on Zitzler-Deb-Thiele's test function. The latter is particularly useful as it allows for both the dimensions of the design space and the size of the solution space to be parametrically modified. In order to assess the relationships between the value of the complexity factor and the benefits of the proposed algorithm, the results are compared to the Pareto frontier that would be provided using a DoE with the same number of function calls. To account for the fact that multiple architectures must be simultaneously optimized, each test function is duplicated and slightly modified. Figures 101 to 103 display the results for an increasing value of the complexity factor due to either an increase in the design space or an increase in the solution space.




(a) Small design space

(b) Small solution space

**Figure 101:** Results for small complexity factors

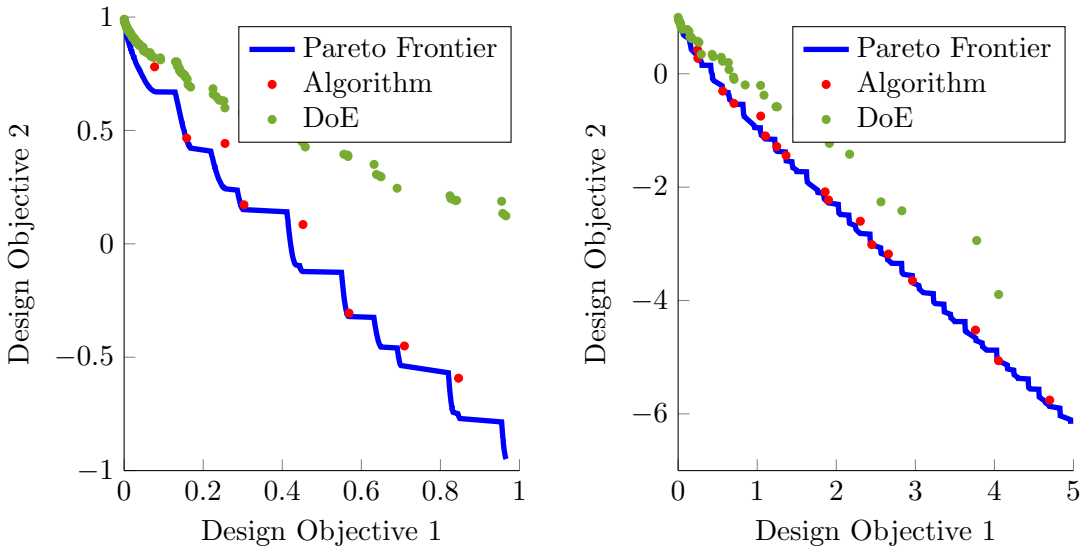



(a) Medium design space

(b) Medium solution space

**Figure 102:** Results for medium complexity factors

As shown in the previous figures, increasing the complexity of the problem highly improves the performance of the proposed algorithm compared to a DoE. The points generated by the proposed algorithm are closer to the real Pareto frontier than the ones generated by the DoE. Moreover, the sampling of the Pareto frontier generated by the algorithm is more




(a) Large design space

(b) Large solution space

**Figure 103:** Results for large complexity factors

evenly spaced compared to the one generated by the DoE, especially for highly complex problems. It can be noted that the performance of the proposed algorithm is not sensitive to the problem's complexity and mainly depends on the selected population size. However, the performance of the DoE is highly deteriorated when the complexity of the problem increases. As a consequence, the proposed algorithm is considered to be a more stable way to determine the Pareto frontier. Finally, as the complexity of current aerospace optimization problems tends to rapidly increase, the benefits and the usefulness of the proposed algorithm are undeniable.

#### *7.2.5.1 Conclusion*

Implementing EMMA has numerous advantages. First, it enables promising architectures to be favored in the selection process. Indeed, an architecture with a large number of non-dominated points on the overall Pareto frontier has a high probability of being further optimized. Moreover, the proposed algorithm still enables less promising architectures to be studied using a probabilistic approach. Indeed, if a given architecture does not have at least one point on the overall Pareto frontier, it still has a probability $p$ of being further optimized. This ensures that all promising configurations have been investigated. As discussed

in this section, the proposed algorithm greatly improves the accuracy of the Pareto frontier sampling, especially when the size of the design space and/or the solution space increases. Indeed, the Pareto frontiers obtained using the proposed algorithm are more evenly spaced and closer to the theoretical ones, especially when the complexity factor increases. These observations result in the validation of the three criteria related to Experiment 1.2:

- Most of the designs from the DoE are dominated by points from the proposed algorithm.
- The proposed algorithm provides an evenly-sampled Pareto frontier.
- The benefits of the proposed algorithm improve with the complexity of the problem.

This leads to the validation of Hypothesis 1.2:

**VALIDATION HYPOTHESIS 1.2: IF an evolutionary multi-architecture multi-objective algorithm based on architecture fitness is developed to drive a sequential use of multiple NSGA-IIs THEN the Pareto frontier of solutions defined by different sets of continuous, discrete, and categorical variables can be efficiently generated.**

## *7.3 Objective Prioritization and Concept Ranking*

The goal of the decision-making environment is not only to provide rapid trade-off analysis capabilities but also to enable a systematic and rigorous selection of the best concept(s) with respect to a given set of design objectives. Therefore, a MADM technique must be used to rank the different concepts according to the stated criteria. As discussed in Section 3.4, the TOPSIS appears to be the most suitable technique for this research [247]. This section first discusses the process used to identify and weigh the importance given to each design objective. Then, it provides the methodology used to rank the concept based on the objective prioritization.

### 7.3.1 Objective Identification and Prioritization

In a complex multi-objective decision space, it is crucial to adequately identify and prioritize the design objectives according to stakeholders' expectations. While surveys are often used in collaboration with social networks [107, 115] to identify the degree of acceptation or satisfaction, there is a need to ask for customers' opinions earlier in the design process and to include them all along the process to decrease cost of change, to improve products, and to have solutions that better fit their expectations. Indeed, in traditional approaches, all the costs incurred become useless and the design process has to start over, leading to higher costs [275]. For that purpose, the stakeholders' point of view is retrieved via a systematic survey that designers can send to potential stakeholders. By adding this stakeholder-centric prioritization, it is ensured that stakeholders' requirements are validated throughout the entire design process. The proposed five-step process is detailed below:

1. Stakeholders are identified and grouped into categories such as consumers, investors, regulators, etc.

2. All stakeholders are asked to grade the potential objectives according to the importance they want to give to each of them. The questions are formulated as follows: "How important is Objective A?". Stakeholders have five possible answers that map to a normalized numerical scale: very important (10), important (5), somewhat important (0), not very important (-5), and not important at all (-10).

3. The average of all grades assigned by stakeholders within each category is then calculated for each objective.

4. The relative importance of each category of stakeholders is determined by decision makers.

5. The weighted average of the relative importance assigned to each objective is calculated based on the relative importance of each objective in each category and the importance of the given category.

Based on these identified and prioritized objectives, concepts are then ranked, as discussed in the next section.

### 7.3.2 Concept Ranking

The TOPSIS ranks alternatives based on their Euclidean distance to the utopia solution. The ranking of all alternatives is then obtained: from the best (closest to the utopia point) to the worst (farthest from the utopia point). The distance is computed using a utility function, as presented in Equation 158 for each concept. In this equation, the weighting factor $w_i$ represents the importance of the i$^{\text{th}}$ design objective, $\bar{y}_i$ represents the normalized grade of the studied alternative with respect to the i$^{\text{th}}$ objective, and $\delta_i$ represents the direction of improvement of the i$^{\text{th}}$ design objective ($\delta_i = +1$ for maximization and $\delta_i = -1$ for minimization).

$$U(y) = w_i \sum_i \bar{y}_i \delta_i \tag{158}$$

For consistency purposes, the values of the design objectives must be normalized. The normalization is made using Equation 159, where $y_i$ represents the grade of the studied alternative with respect to the i$^{\text{th}}$ objective, $y_{min}$ the minimum value of the i$^{\text{th}}$ objective, and $y_{max}$ the maximum value of the i$^{\text{th}}$ objective.

$$\bar{y}_i = \frac{y_i - y_{min}}{\max\left(y_{max} - y_{min}, 10^{-5}\right)} \tag{159}$$

This technique allows designers to develop multiple scenarios by changing their design priorities and assessing the impact of this change on the best(s) concept(s).

In order to account for uncertainty in design priorities, a three-step approach that selects the most robust design(s) is developed:

1. Uncertainty definition: instead of using a single values for the multiple design priorities, probability distributions are developed. In particular, Gaussian distributions, characterized by both the most probable value and the standard deviation are implemented.

2. Uncertainty propagation: to propagate uncertainty in the process, a Monte Carlo simulation is used. The latter samples the previously modeled Gaussian distributions for each variable in order to produce a large number of possible sets of priorities, also called scenarios. For each scenario, a ranking of alternatives is obtained using the aforementioned TOPSIS.

3. Concept selection: the concept selection is performed by outputting a final ranking. The latter is calculated by averaging the rank of each alternative over all generated scenarios.

This approach provides a probabilistic way to model, propagate, and include uncertainty in the design priorities. As a consequence, not only has the best concept optimal performance but it also is robust to changes in design priorities.

## *7.4 Visualization*

When designing new unconventional vehicles such as suborbital vehicles, designers usually face a lack of historical data. Since designs can rapidly become overwhelming, the designers' ability to fully understand the problem to be solved can be limited. According to Kamdar et al., "a very large number of variables and physics behind the system are too profound or esoteric to be fully understood" [228]. This might result in the design of non-optimal concepts and consequently in the loss of both time and money [232]. To overcome this difficulty, visualizing the generated concepts would bridge the gap between the information provided by the optimization algorithm and the human cognitive and perceptual systems [279]. In addition, visualization capabilities facilitate the integration of designers' past experience, knowledge, and cognitive capabilities in the analysis. It also supports a better, easier, and faster decision-making process. Designing complex vehicles subject to uncertainty in requirements and a large design space usually involves multi-disciplinary teams. Collaboration and communication within each team and between teams are crucial to the success of the program. The people involved in the projects usually have different backgrounds and work experiences so they tend to have their own specific interpretation and perception of the problem. Hence, having a visualization tool might help reach a

consensus and limit potential misunderstandings and conflicts during the decision-making process [197]. In particular, the visualization module has to provide parametric visualization capabilities as it is necessary to avoid re-iterating on the design process. According to De Beats, it is important to have "a parametric model of the airplane to allow quick changes in the shape and size of the vehicle" [109].

Nevertheless, the visualization module cannot be limited to a communication or presentation tool. It has to be considered as a way to explore and understand the structure of the design space [450]. The data inputted into the visualization module should enable the identification of relationships, trends, clusters, etc. [231]. In addition, according to Meyer et al. [294], "visualization of information alone does not provide new insight". Hence, in order to make informed decisions, a parallel between the systematic optimization process and the human cognitive system has to be made to further sharpen the human reasoning and the knowledge synthesis. An interactive platform would thus present the optimization process results and allow designers to automatically visualize the corresponding CAD model. Designers would therefore have access to the original data in order to step back and interpret the data from multiple angles in order to get the big picture of the problem. By following the given process, potential problems and new perspectives can be highlighted, bringing out new types of innovative solutions.

Allowing decision makers to visualize the various generated concepts becomes a key enabler of the design space exploration of unconventional concepts. In order to facilitate the link between the visualization module and the optimization platform and to provide an ergonomic platform to decision makers, a user interface is developed in Excel. This dashboard is directly fed with the optimization results and drives the visualization based on concepts selected from either the ranking or the Pareto frontier. This section discusses the development of the visualization platform, the CAD model, and their integration into the vehicle.

While the optimization process provides a description of the vehicle using a list of design variables, it is difficult to have a quick overview and understanding of the optimum solution(s). In addition, designers might not be able to easily assess the differences between

two different architectures. In order to bridge the gap between the theoretical optimization process and the decision-making process, there is a need for rapidly visualizing the various solutions proposed.

As a consequence, an extended visualization environment is proposed in association with the design framework. The visualization environment aims at converting the outputs of the optimization process into a high-level CAD model of the vehicle. It facilitates the understanding of the technical solutions and their presentation to customers and decision makers by providing a 3D view. It also gives insight into the actual shape and size of the vehicle. Finally, it allows designers to check the consistency and the feasibility of the obtained concept(s). The proposed visualization dashboard is discussed in the next section.

### 7.4.1 Visualization Dashboard

The visualization dashboard, presented in Figure 104, allows decision makers to drive the visualization in the CAD software. Users can select the concept(s) of interest from either a ranking based on the prioritization algorithm or from the Pareto frontier.

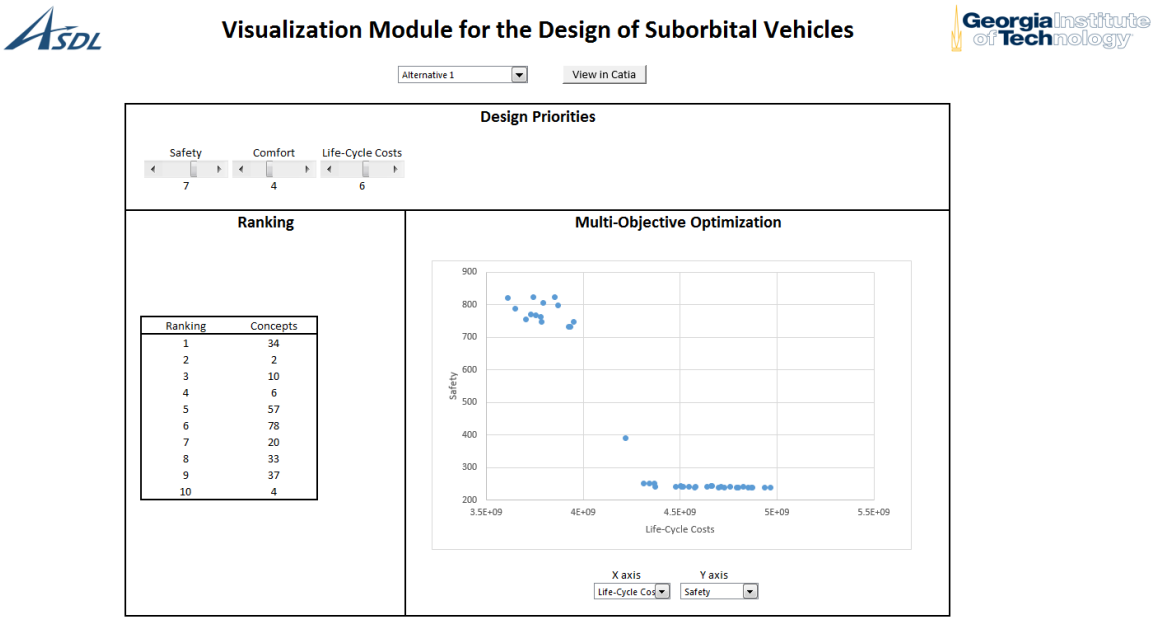



**Figure 104:** Dashboard of the visualization tool

Slider bars are present to provide decision makers with convenient ways to prioritize their

objectives. Changes in priorities automatically update the ranking of the top 10 concepts. In addition, a parametric Pareto frontier is developed so that decision makers can select the metrics among which they want to perform trade-off analyses. This aims at providing decision makers with more flexibility when performing trade-offs. Once the concept of interest has been selected for visualization, a Visual Basic for Applications (VBA) function selects the corresponding values in the Excel file generated by the optimization algorithm and outputs them in the Excel file linked to the CAD model. The development of the parametric CAD model is discussed in the next section.

### 7.4.2 Parametric CAD Model

In order to facilitate decisions and trade-off analyses, all possible configurations must be modeled. Hence, a method needs to be developed that enables each architecture to be modeled and its design variables parametrically changed. For that purpose, a flexible and parametric CAD model is developed. To cover all potential alternatives, the vehicle is broken down into physical components, as described by the tree presented in Figure 105.

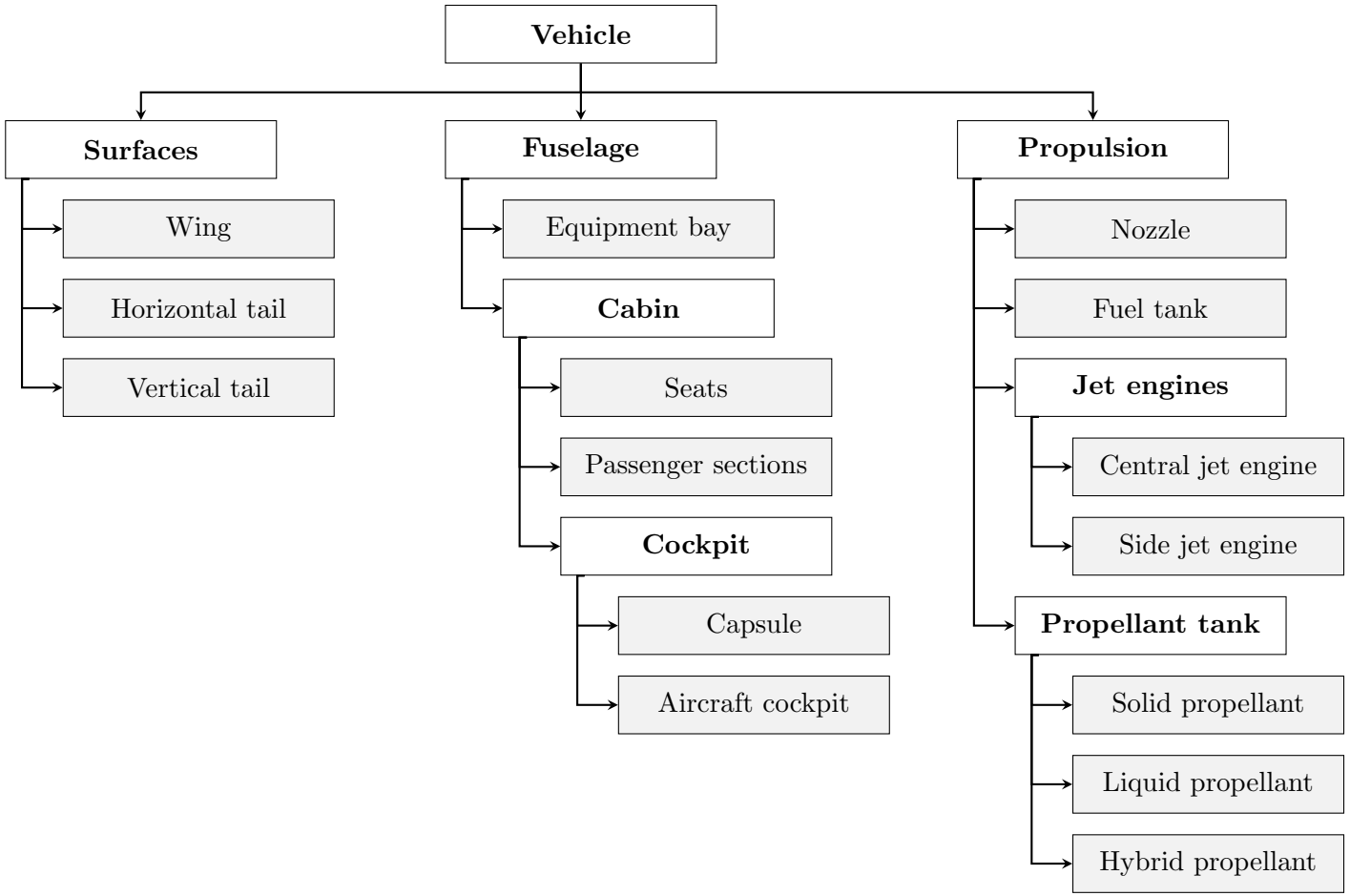



**Figure 105:** Physical breakdown of the vehicle

To develop the different components, a CAD software suite must be chosen. For that purpose, a survey is performed to identify the most commonly used software in the aerospace industry, as presented in Table 48.

**Table 48:** CAD software suites used by the main aerospace company [106, 348, 396]

| **Aerospace companies** | **CAD software suites used** |
|---|---|
| Boeing | Catia |
| Airbus | Catia |
| NASA | Siemens NX, PTC Creo |
| Space X | Siemens NX |
| Lockheed Martin | PTC Creo, Catia |
| Northrop Grumman | PTC Creo, Siemens NX |
| Bombardier | Catia |
| General Dynamics | Siemens NX |
| Pratt & Whitney | Siemens NX, Catia |
| Jet Propulsion Laboratory | Siemens NX |
| Thales Alenia Space | Catia |

As shown in Table 48, two CAD software are mainly used: Catia developed by Dassault Systemes and NX developed by Siemens. Both meet all research requirements for the visualization module and are able to link architecture parameters provided by Matlab to CAD geometric parameters via an Excel file. Catia is chosen for the following reasons:

- Catia licenses are available at the ASDL.
- Contrary to NX, Catia is used by both American and European companies, which makes it more generic.
- Most designers are familiar with Catia and could use the provided model to adjust and adapt the proposed vehicle to their specific geometric requirements.

Thus, the various components will be modeled in Catia, as described in Appendix C.

### 7.4.3 Integration

Once each component has been modeled, boolean logic is used to create feasible concepts. In addition, the vehicle parameters can be varied in order to model every vehicle of interest

generated by the optimization algorithm. Three representative vehicles are provided in order to visualize all architectures:

- Winged vehicle with jet engines (Figure 106)
- Winged vehicle without jet engines (Figure 107)
- Slender vehicle without jet engines (Figure 108)

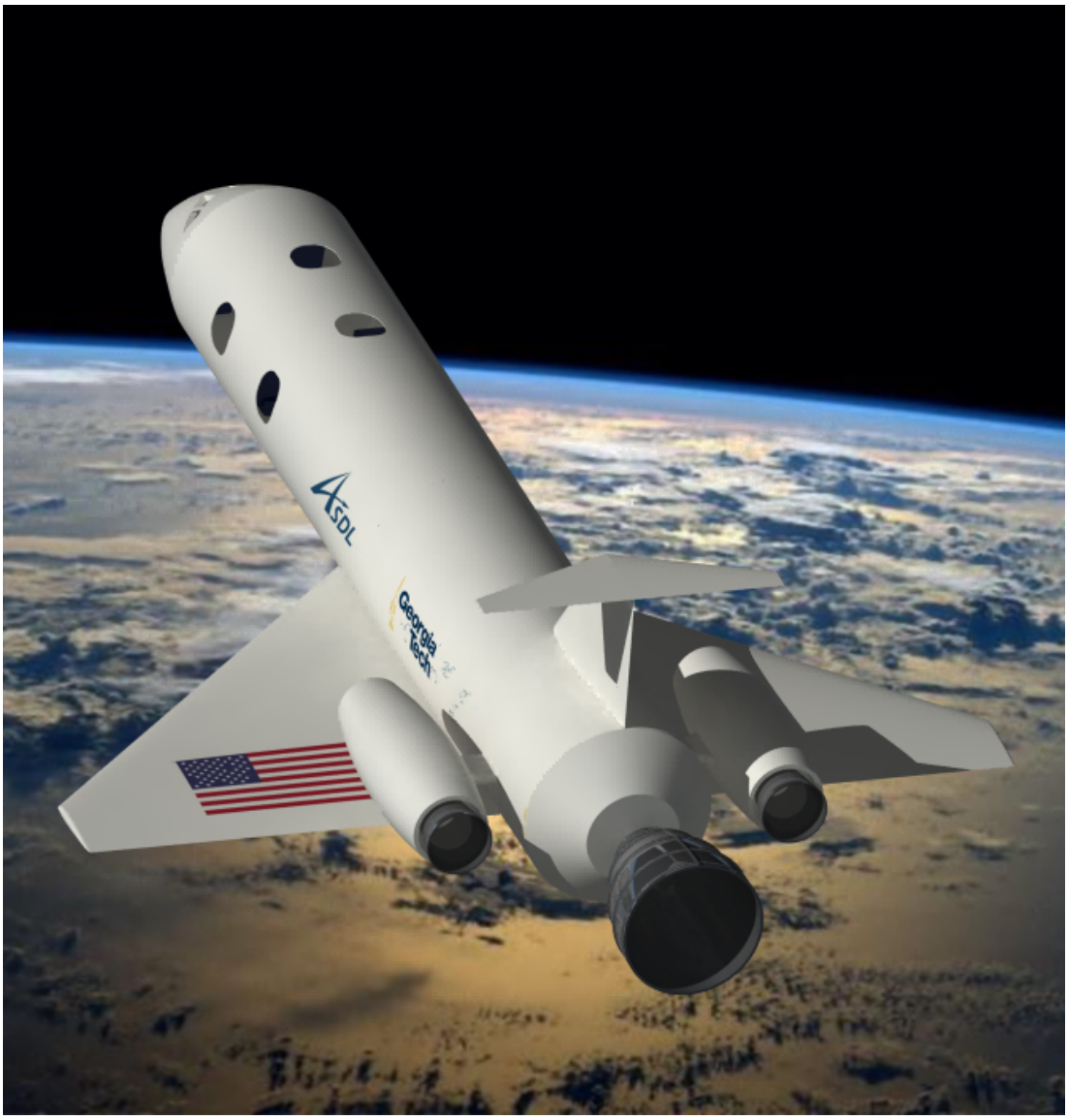



**Figure 106:** Winged vehicle with jet engines


**Figure 107:** Winged vehicle without jet engines


**Figure 108:** Slender vehicle without jet engines

This chapter discussed the development of the last step of the proposed methodology, which consists in providing decision makers with the capabilities required to make informed decisions. Hence, all the pieces of the proposed methodology have been discussed in Chapters 4 to 7 and are summarized in the next chapter.

# CHAPTER VIII

# SUMMARY OF THE PROPOSED METHODOLOGY

This chapter aims at summarizing the key elements of the methodology developed in Chapters 4 to 7 that support the Overarching Hypothesis formulated in Section 2.3:

**IF a variable-oriented morphological analysis is used to feed an evolutionary multi-objective multi-architecture optimization algorithm AND IF fuzzy set theory is used to parametrically propagate requirements' uncertainty through a multi-disciplinary physics-based modeling and simulation environment THEN large multi-architecture design spaces can be better explored AND informed decisions can be made under evolving uncertainty in requirements.**

The proposed methodology is decomposed into four main steps that follow the generic top-down design decision support process:

- Step 1: Establish the decision criteria
- Step 2: Define the design space
- Step 3: Evaluate alternatives
- Step 4: Make informed decisions

The relationships between the key elements of each step are provided in Figure 109, which summarizes the proposed methodology named ASCEND: Architecture Selection under multiple Criteria and Evolving Needs for improved Decision-making. Step 1 aims at identifying the metrics of interest, which can be decomposed into two categories: constraints and objectives. During the optimization process, constraints are used to limit the design space and objectives are used to formulate the objective functions of the optimization algorithm. In addition, priorities are defined along with their uncertainty for further objective prioritization. Step 2 aims at generating all feasible alternatives, which can be used for

further comparison and optimization. For that purpose, the system to be designed is decomposed into functions, and options are identified for each function. Then, all feasible alternatives are generated and design variables are infused in the process. This enables alternatives to be grouped into variable-oriented architectures. Those architectures, along with their design variables, are inputted into Step 3, which aims at evaluating all possible concepts. Architectures can be evaluated using different modeling and simulation environments, depending on the complexity of the problem, the level of fidelity required, and the availability of existing design frameworks. The latter are used in Step 4, which supports the overall decision-making process. For that purpose, the process optimizes the various architectures against multiple criteria. It uses an evolutionary algorithm, which drives parallel multi-objective optimization algorithms. Once the overall Pareto frontier is obtained, non-dominated concepts are ranked based on a probabilistic TOPSIS and are used to perform important trade-offs between the various technological solutions. In addition, informed decisions are supported by a visualization dashboard, which enables rapid and parametric visual trade-offs among the best concepts.

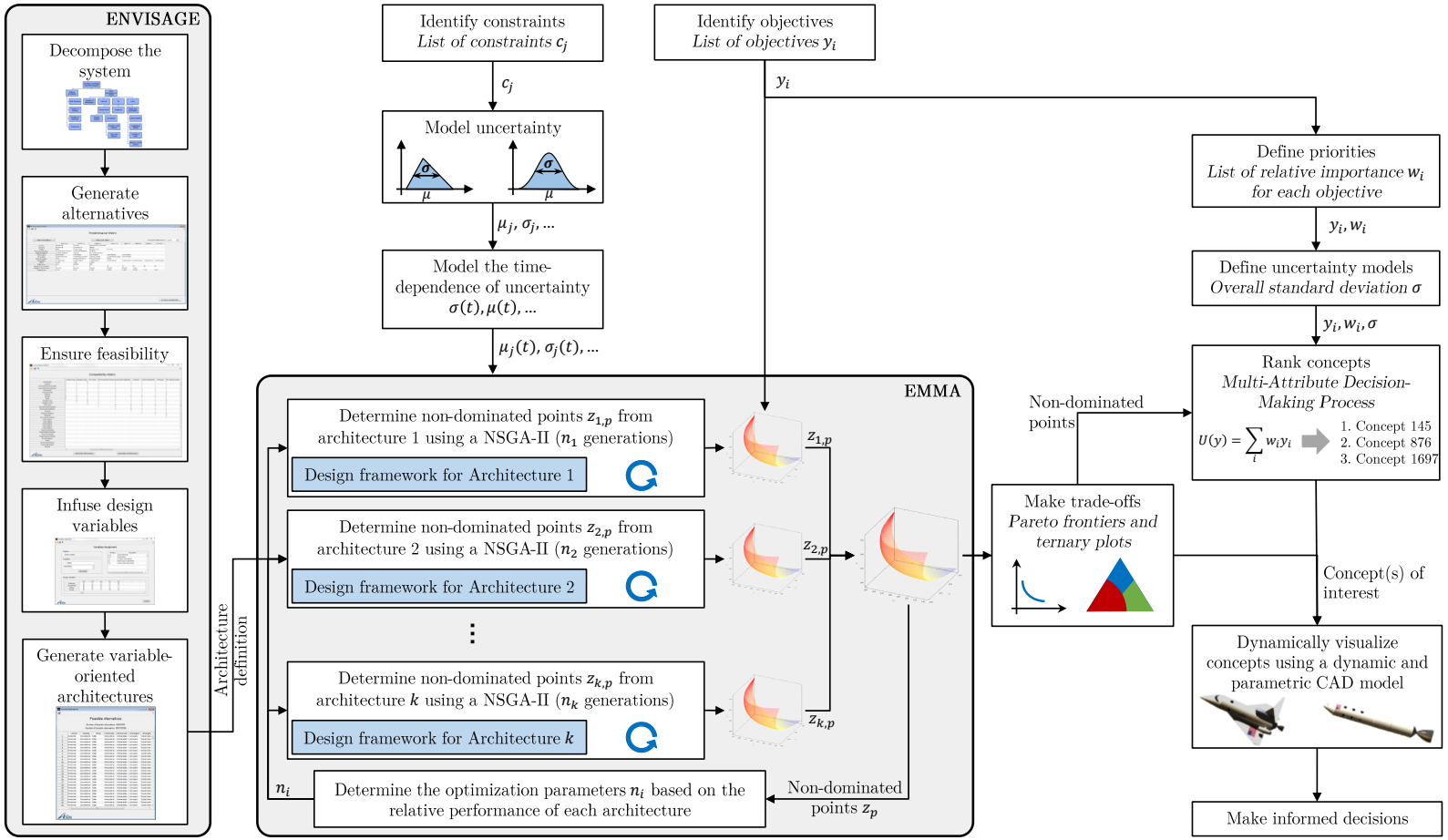



**Figure 109:** Overview of ASCEND

The following chapter further discusses the validation of the Overarching Hypothesis by implementing ASCEND and applying it to the design of a profitable, safe, and robust suborbital program.

# CHAPTER IX

# CLOSING THE LOOP

This chapter aims at closing the loop of the overall research. Chapter 1 identified key assertions that resulted in the establishment of the following two research objectives:

- To enable a broad and informed design space exploration for optimized vehicle selection at a conceptual design level.
- To support decisions under unclear objectives and evolving uncertainty in requirements.

Based on these two research focuses, the following overarching research objective has been formulated:

**To establish a methodology that enables a broad design space exploration at a conceptual level to select solutions against unclear objectives and under evolving uncertainty in requirements.**

The present chapter focuses on the validation of the proposed methodology by comparing its performance against each of the three existing approaches identified in Section 2.1:

- Typical aircraft design approach
- Architecture comparison approach
- Architecture optimization approach

For that purpose, key validation criteria are first established. Then, experiments are developed and set up. Based on the results of these experiments, the validation criteria established are evaluated for the proposed methodology as well as for the other approaches. Finally, the performance of the proposed methodology is benchmarked against the other approaches.

## *9.1 Establishment of the Validation Criteria*

The objectives of the proposed methodology are both to improve the exploration of large design spaces and to support informed decisions under evolving uncertainty in requirements. In order to test and validate the Overarching Hypothesis, a list of criteria must be established that enable a methodology to be evaluated with respect to both objectives, as discussed in this section.

### 9.1.1 Objective 1: Exploring Large Design Spaces

Four criteria can be established to evaluate the ability of the proposed methodology to support the exploration of large design spaces.

**Criterion 1: Alternatives evaluated with different modeling and simulation environments can be systematically compared and optimized.** When solving transportation problems, multiple architectures must be considered, modeled, optimized, and compared. This is especially true for new types of missions or markets such as suborbital tourism and flying cars for which experience is rare. As a consequence, it is crucial to handle new complex design problems using a systematic, rigorous, and traceable approach that has the ability to ensure that the best solution is found. To avoid sub-optimal solutions or missing opportunities, bias from expert judgment must be avoided. For that purpose, all possible configurations must be considered during the design process. In order to consider all possible design alternatives, the capability to evaluate their performance must be developed. However, there is a natural trade-off between breadth and depth in the evaluation process; while a modeling and simulation environment based on first principles is usually able to handle a large portfolio of concepts, its modeling accuracy is poor. In contrast, CFD or Finite Element Method (FEM) codes tend to be extremely accurate but specific to a given configuration or set of concepts. Hence, increasing the fidelity of the models requires designers to use an increasing number of modeling and simulation environments. However, as demonstrated in Chapter 2, there is a lack of methodology that enables a systematic comparison and optimization of concepts modeled by different modeling and simulation

environments.

**Criterion 2: The proposed methodology improves the total computational time without reducing the number of alternatives considered.** Accounting for all design alternatives results in the evaluation of an extremely large number of concepts. For instance, the high-level decomposition of suborbital vehicles performed in Section 5.3 shows that the number of alternatives to be considered rapidly increases and can easily reach several millions. In addition, improving the fidelity of the modeling and simulation environment tends to increase the time it takes to evaluate each alternative. When using a morphological analysis to identify the possible options, Equation 160 can be used to compute the total time $T_t$ required to evaluate all alternatives, where $t_s$ is the average time of a single concept evaluation, $N_f$ the number of functions, and $n_i$ the number of options for the i$^{\text{th}}$ function.

$$T_t = t_s \prod_{i=1}^{N_f} n_i \tag{160}$$

Assuming an average number of options for each function equal to five, Table 49 provides the total required evaluation time for different combinations of number of functions and concept evaluation time.

**Table 49:** Execution time (years) for various problem sizes

| | | **Single alternative execution time (s)** | | | | |
|---|---|---|---|---|---|---|
| | | **1** | **5** | **10** | **15** | **20** |
| **Number of functions** | **8** | 0.01 | 0.06 | 0.12 | 0.19 | 0.25 |
| | **10** | 0.31 | 1.55 | 3.10 | 4.45 | 6.19 |
| | **15** | 968 | 4,838 | 9,677 | 14,516 | 19,354 |
| | **20** | 3 million | 15 million | 30 million | 45 million | 60 million |
| | **25** | 9 billion | 47 billion | 95 billion | 142 billion | 189 billion |

As described in Table 49, the total time required to evaluate all possible alternatives rapidly reaches impractical values. Even for the simplest problems, several days are required to explore all alternatives. As a consequence, reducing the execution time is critical to improve current design space exploration approaches.

**Criterion 3: The proposed methodology provides a better coverage of the design space than current design space exploration approaches.** The ultimate goal of improving the design space exploration is to generate the best possible designs. While the notion of "best designs" might be complicated to define in a multi-objective environment, some metrics can be established to measure how well the design space is covered. One of such metrics is the average distance of the points on the Pareto frontier compared to the ideal solution. The ideal solution is defined as the combination of the best values for each objective. Hence, the smaller the average distance, the better the coverage. This average distance can be calculated using Equation 161, where $k$ is the number of points on the Pareto frontier, $y_{i,m}$ the value of the i$^{\text{th}}$ point with respect to each objective $m$, $y_m^*$ the best possible value of each objective $m$, and $y_m^-$ the worst possible value of each objective $m$.

$$\delta^* = \sum_m \frac{1}{k} \sum_{i=1}^{k} \frac{|y_{i,m} - y_m^*|}{|y_m^* - y_m^-|} \tag{161}$$

In addition, the number of non-dominated concepts can also be used as an evaluation criterion: the larger the number of non-dominated points, the better the coverage. Finally, for a fixed set of priorities, the performance of the best concept can be evaluated and used as an evaluation criterion: the better the performance of the vehicle, the better the coverage. Hence, Criterion 3 can be quantified using the following three metrics:

- The average distance between the points on the Pareto frontier and the ideal solution.
- The number of non-dominated points in the solution space.
- The performance of the vehicle optimized for given sets of design priorities.

**Criterion 4: The methodology provides the capabilities to both perform**

**quantitative trade-offs and identify key trends among all alternatives.** As previously mentioned, systems tend to become more and more complex and current economic conditions result in a shortened time-to-market and an increasing demand for flexibility and adaptability. However, the difficulty of making good decisions does not only stem from the aleatory aspect of the performance of a vehicle or the health of the market. The multiplicity of objectives and disciplines involved in the full life-cycle of complex aerospace vehicle programs complicates the task of key decision makers. In order to build an optimized and profitable suborbital program, designers do not only need to consider flying performance, but also economic, safety, and robustness metrics. Usually, most of these objectives are concurrent, and therefore, cross-functional trade-offs have to be made. In addition, in early phases of a market, some technologies are still in development. For that reason, there is a need to identify the most promising technologies and key design drivers in order to align and optimize the budget allocated to research and development departments.

The validation of the four aforementioned validation criteria would enable the validation of the part of the research objective related to the design space exploration. The following section focuses on assessing the methodology's performance with respect to its ability to support key decisions under evolving uncertainty in requirements.

### 9.1.2 Objective 2: Supporting Informed Decision-Making Under Evolving Uncertainty in Requirements

Three additional criteria can be established to assess the ability of the proposed methodology to support informed decision-making under evolving uncertainty in requirements.

**Criterion 5: The proposed methodology provides decision makers with a complete picture from both a physical and a business standpoints, hence supporting informed decision-making.** Large and innovative aerospace endeavors such as space tourism tend to present both an important potential and a growing interest from the general public. While they might be highly profitable, such complex aerospace programs are considered to be among the most costly and risky projects [316]. Indeed, a lot of money has to be committed to the development of a single vehicle. In addition, such programs are

exposed to a broad range of significant risks (customer requirements, demand, etc.), and to potentially harsh competition. They also often have a late payback period and require a large amount of funding. Moreover, as discussed in Section 1.3, requirements' uncertainty tends to be extremely present, especially at the dawn of emerging markets. While this uncertainty tends to decrease over time, there is a need to support well-founded decisions under evolving uncertainty in requirements.

One of the key characteristics of informed decisions is their ability to be supported by quantifiable and pertinent metrics at each point in time. Three metrics help alleviate the bias provided by the designers' experience and consequently avoid missing any opportunities. In addition, quantifying some of the decision metrics allows for more rational and traceable decisions. This also allows for results to be concisely and efficiently presented to upper management and key stakeholders. Finally, with quantitative information, it is possible to manipulate the information in consistent and reproducible ways, by combining figures, comparing data, examining rates of change, etc., hence improving the accuracy of the decisions made.

As discussed in Section 7.4, designing complex vehicles subject to uncertainty in their requirements and a large design space usually involves multi-disciplinary teams. Collaboration and communication within each team and between teams are crucial to the success of the program. Hence, having a visualization environment might help reach a consensus and limit potential misunderstandings and conflicts during the decision-making process [197]. In particular, the methodology has to provide parametric visualization capabilities in order to avoid having to re-iterate on the design process. Finally, a visualization environment is usually necessary to check the consistency of the design provided by the optimization algorithm.

**Criterion 6: The methodology provides the ability to easily trade performance against robustness to alleviate the program risk.**

It is common that for emerging markets the regulatory frameworks necessary to informed decision-making in general and the concept selection in particular are usually not fully defined at the early development stages. For instance, suborbital flights are performed by

either an aircraft or a spacecraft but independently of the concept used, they need to reach the limit between the atmosphere and outer space. Therefore, an important question is naturally raised: should the space law, the air law or even both laws be applied? The answer to this question will have significant legal implications on suborbital space tourism. Indeed, as mentioned by Axelle Cartier and Ioana Cristoiu [71], "suborbital flights are at present the major issue to be dealt with under the current legislation. [...] It is not clear what will be the applicable law. It is to be expected that issues will depend on different national legislations." Moreover, this legal framework is not complete yet and will continuously evolve with the introduction of new vehicles, new countries and new companies [143, 252]. Also, at the dawn of a new vehicle development, its applications might not be clearly defined. Similarly, customers' requirements are extremely uncertain, especially in early phases. Therefore, in order to support informed decisions and alleviate the program's risk, the methodology needs to identify designs that are robust to changes in requirements, market constraints, etc.

**Criterion 7: The methodology generates the information required to support informed go/no-go decisions.**

While the degree of uncertainty tends to be high at the early stages, it tends to decrease with the progressive establishment of the regulatory framework and the identification of potential applications. Hence, decision makers constantly face an important dilemma: freeze the design and start the program or continue the development and wait until more information becomes available about the market. The first strategy allows a firm to benefit from the first-mover advantage. However, the probability of developing an optimized and successful design is lower because of the lack of knowledge about the requirements and the market. In contrast, the second strategy reduces the program risk but also reduces the potential NPV due to the time value of money: money now is worth more than money in the future. As a consequence, there is a need for the methodology to support decision makers navigate such dilemmas.

In order to evaluate and validate each of the aforementioned criteria, an experiment

needs to be set up, as discussed in the following section. The goal is to generate results that will be used to assess each criterion, and consequently validate the Overarching Hypothesis.

## *9.2 Experiment Setup*

The proposed methodology aims at improving the development of emerging markets by supporting both the exploration of large design spaces and decisions under evolving uncertainty in requirements. As discussed in Chapter 1, suborbital vehicles are representative of such emerging markets. Consequently, they are used as a test bed to validate the Overarching Hypothesis. In particular, the objective of the experiments is to assess the ability of the proposed methodology to meet the aforementioned criteria and consequently support decision makers in developing a profitable suborbital program. This section first provides an overview of the different steps of the experimental apparatus. Finally, the implementation of each of these steps is detailed.

### 9.2.1 Overview of the Experimental Apparatus

The experimental apparatus is based on the methodology discussed in Chapter 8 and displayed in Figure 110. The following sections discuss how the relevant experiments are set up within the context of each of the steps of the proposed methodology: establishment of the decision criteria, definition of the design space, alternative evaluation, and decision-making.

### 9.2.2 Step 1: Establish the Decision Criteria

While numerous potential decision criteria have been identified in Chapter 4, this experiment focuses on safety, NPV, and passenger experience. Indeed, those three metrics will ensure that the program is compliant with the regulations, while being attractive, and economically viable. It is assumed that all the phases of the vehicle life-cycle (research and development, manufacturing, and operations) are handled by a single company.

Five uncertainty sources are considered: maximum acceptable load factor, maximum altitude reached by the vehicle, demand, duration of the development phase, and risk of catastrophic failure. The models developed for the first four sources of uncertainty are provided in Table 50. In order to reduce the computational time, the membership functions

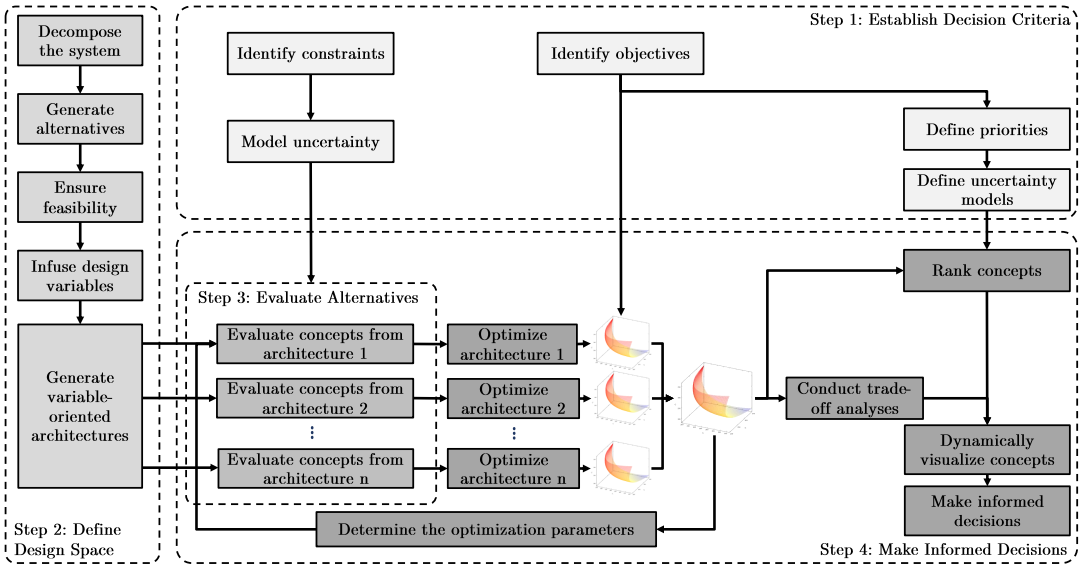



**Figure 110:** Overall methodology

are modeled by only parameters: the mean value and the standard deviation. In addition, only the standard deviation is modeled with a time-dependent scaling factor.

**Table 50:** Model of the various uncertainty sources

| **Parameters** | **Maximum altitude** | **Maximum load factor** | **Demand** | **Development time** |
|---|---|---|---|---|
| **Distribution** | Trapezoidal | Trapezoidal | Triangular | Triangular |
| **Mean** | 100 km | 4.5 g | 15,000 passengers | 6 years |
| **Standard deviation** | 16 km | 0.3 g | 4,000 passengers | 0.5 year |
| **Time-dependence** | Exponential with a 3-year half life | Exponential with a 3-year half life | Exponential with a 10-year half life | Constant |

Figure 111 displays the notional evolution of the standard deviation over time for a three-year and a ten-year half life evolution. As shown, uncertainty distributions with a three-year half life only have a 10% uncertainty after ten years, while the ones with a ten-year half life still have 50% of their uncertainty at that time.

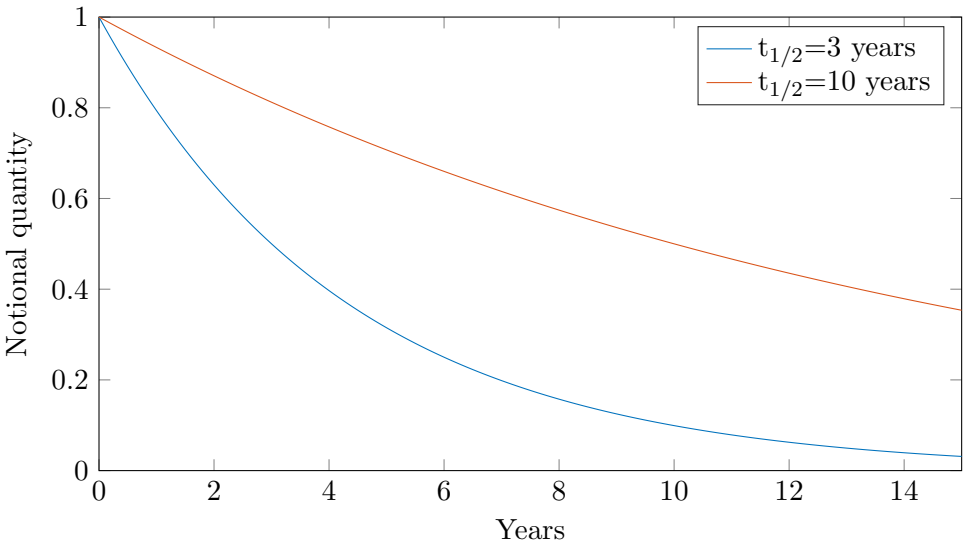



**Figure 111:** Evolution of the different standard deviations over time

The risk of catastrophic failure is modeled as a simple event with a probability of occurrence $p_f$, which is a function of the risk level. It results in a standard deviation $SD_f$ on the NPV as described in Equations 162 and 163, where $R_L$ is the risk level of the vehicle and $C_f$ the impact of the failure on the NPV [313].

$$p_f = \frac{1}{10^5} R_L \tag{162}$$

$$SD_f = \sqrt{p_f\,(1 - p_f)} C_f \tag{163}$$

Finally, it is assumed that all variables that are subject to uncertainty are independent so that their covariances are equal to zero. As a consequence, Equation 148, which is used to propagate uncertainty through the design framework, can be simplified and rewritten as Equation 164, where $\sigma_i^2$ is the variance of $x_i$ and $g_i$ the partial derivative of $f$ with respect to $x_i$.

$$\mathrm{Var}(X) = \sum_{i=1}^{n} g_i^2(\mu)\sigma_i^2 \tag{164}$$

### 9.2.3 Step 2: Define the Design Space

Based on the methodology described in Chapter 5, the definition of the design space follows a five-step process that will be detailed in this section. Once the system has been decomposed, the software ENVISAGE is used to define all possible options, ensure feasibility, infuse design variables, and generate feasible variable-oriented architectures.

#### *9.2.3.1 Decompose the System*

The system decomposition consists in breaking down the main objective into functions.

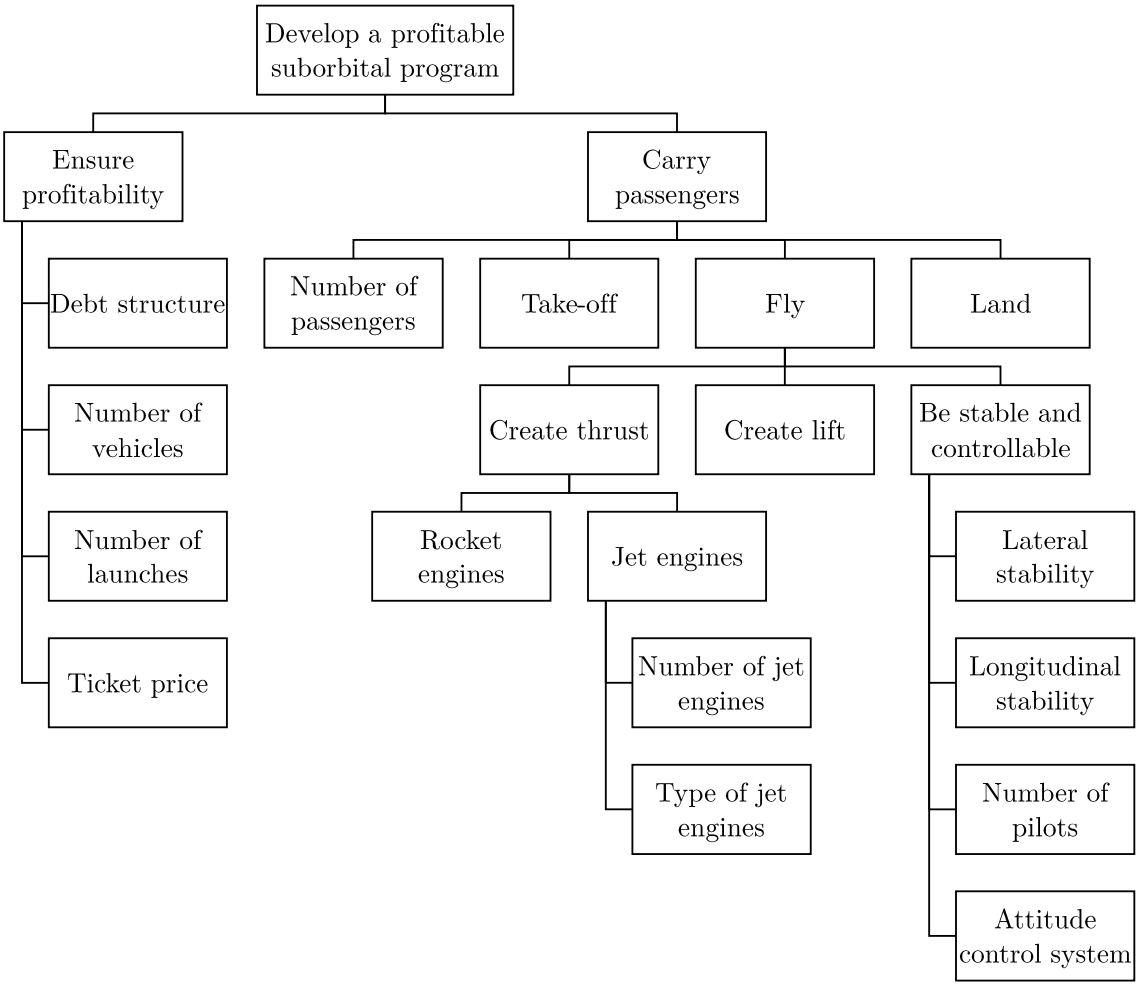
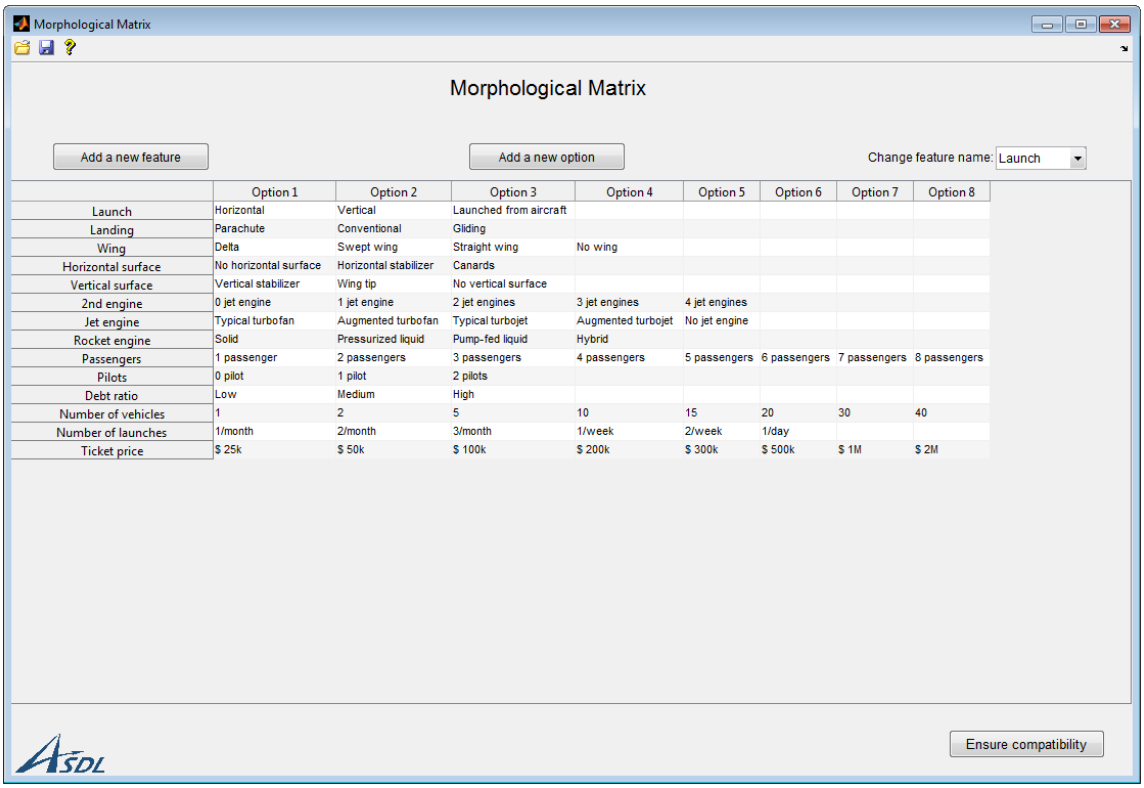



**Figure 112:** Functional decomposition of the system

For this experiment, the main objective is to develop a profitable suborbital program. This objective is then broken down into two main sub-objectives: to carry passengers and to ensure profitability. Figure 112 displays the detailed functional decomposition of the problem based on the analysis performed in Section 5.3.

#### *9.2.3.2 Define Possible Options*

The different options specific to each function can be easily identified through a morphological analysis. Figure 113 shows the morphological matrix created using the software ENVISAGE. Compared to the work done in Section 5.3, some options have been removed to match the capabilities of the developed design framework. New program and business-related features such as the number of vehicles used, the number of launches per month, the debt ratio, and the ticket price have also been added. The updated morphological analysis opens a design space of around 900 million possible combinations.




| | Option 1 | Option 2 | Option 3 | Option 4 | Option 5 | Option 6 | Option 7 | Option 8 |
|---|---|---|---|---|---|---|---|---|
| Launch | Horizontal | Vertical | Launched from aircraft | | | | | |
| Landing | Parachute | Conventional | Gliding | | | | | |
| Wing | Delta | Swept wing | Straight wing | No wing | | | | |
| Horizontal surface | No horizontal surface | Horizontal stabilizer | Canards | | | | | |
| Vertical surface | Vertical stabilizer | Wing tip | No vertical surface | | | | | |
| 2nd engine | 0 jet engine | 1 jet engine | 2 jet engines | 3 jet engines | 4 jet engines | | | |
| Jet engine | Typical turbofan | Augmented turbofan | Typical turbojet | Augmented turbojet | No jet engine | | | |
| Rocket engine | Solid | Pressurized liquid | Pump-fed liquid | Hybrid | | | | |
| Passengers | 1 passenger | 2 passengers | 3 passengers | 4 passengers | 5 passengers | 6 passengers | 7 passengers | 8 passengers |
| Pilots | 0 pilot | 1 pilot | 2 pilots | | | | | |
| Debt ratio | Low | Medium | High | | | | | |
| Number of vehicles | 1 | 2 | 5 | 10 | 15 | 20 | 30 | 40 |
| Number of launches | 1/month | 2/month | 3/month | 1/week | 2/week | 1/day | | |
| Ticket price | $ 25k | $ 50k | $ 100k | $ 200k | $ 300k | $ 500k | $ 1M | $ 2M |

**Figure 113:** Morphological matrix created with ENVISAGE

#### *9.2.3.3 Ensure Feasibility*

The compatibility between options is set based on the authors' knowledge and judgment. For example, a slender body cannot take off or land horizontally. In addition, there is no incentive to carry jet engines if the vehicle is launched from an aircraft or from a balloon. Other similar considerations are made to build the compatibility matrix (Figure 114). Once completed, ENVISAGE is executed in order to extract all feasible alternatives.

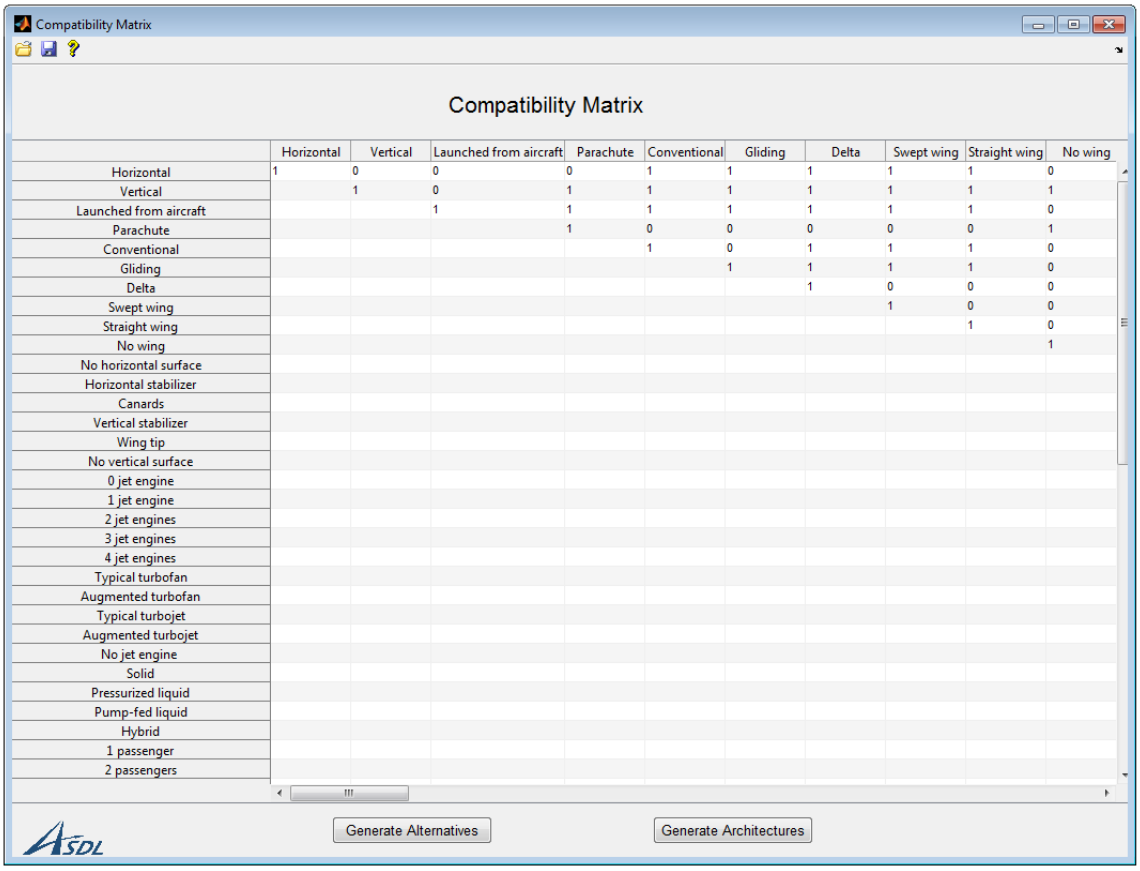



**Figure 114:** Compatibility matrix created with ENVISAGE

A list of around 47 million feasible alternatives is provided, as displayed in Figure 115. Accounting for the compatibility between options results in dividing the number of alternatives to be investigated by around 20.

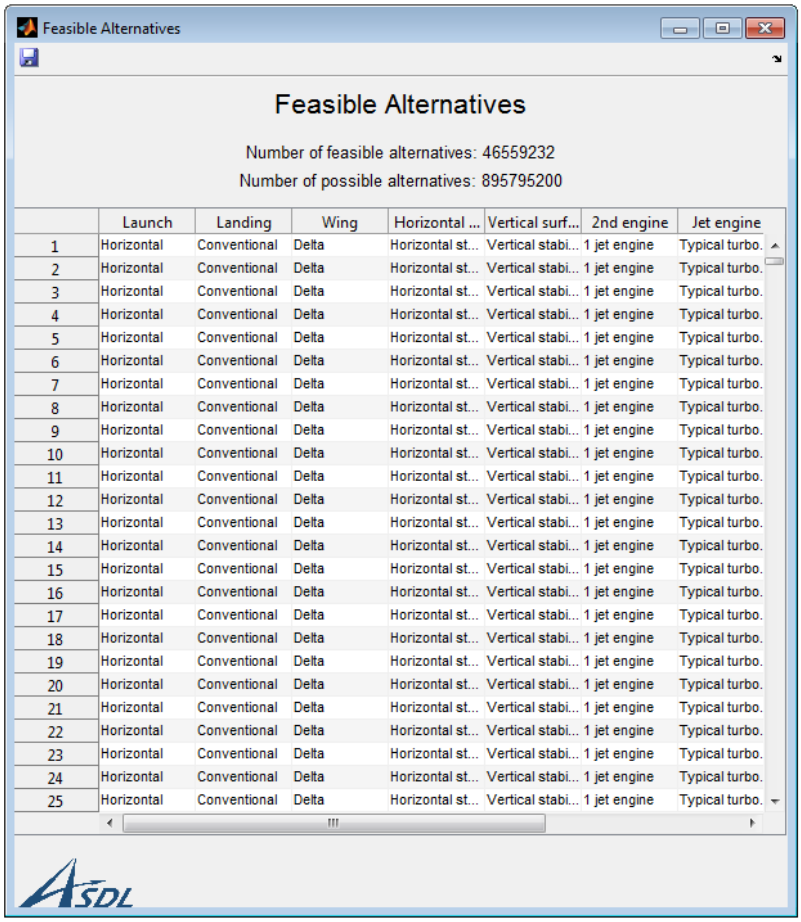



**Figure 115:** List of feasible alternatives generated with ENVISAGE

#### *9.2.3.4 Infuse Design Variables*

This step aims at defining the design variables useful to define each function and to allocate these variables to their specific option. Figure 116 displays the allocation of the wing variables to the different options. The same process is repeated for each function.

#### *9.2.3.5 Generate Feasible Variable-Oriented Architectures*

Once the design variables are listed and assigned to the corresponding options, ENVISAGE is executed and four feasible architectures are identified. A summary of the architecture generation process is provided in Figure 117. Starting with about 900 million total combinations, the integration of the compatibility matrix within the process allows designers to divide the number of alternatives to be investigated by a factor of 20. Grouping

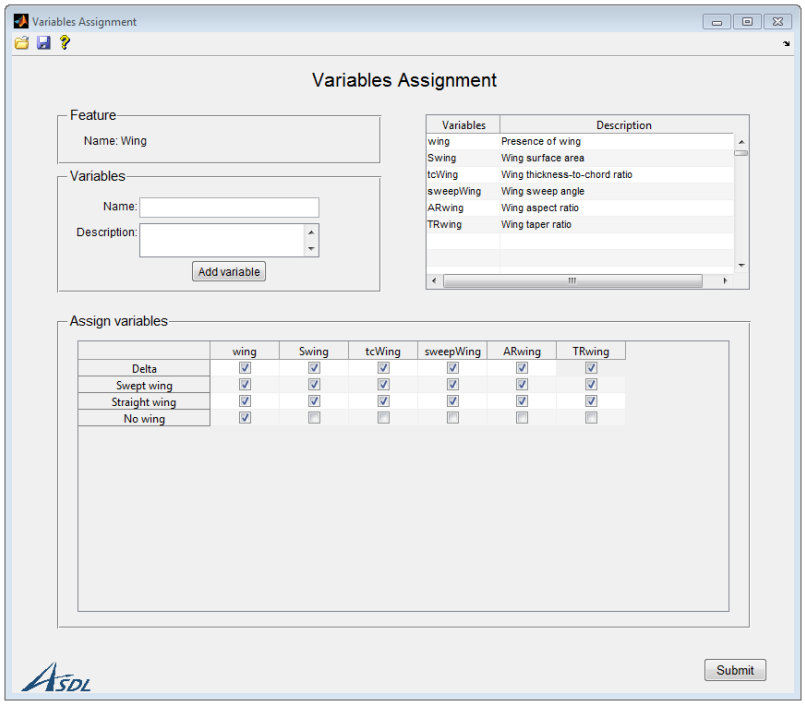



**Figure 116:** Infusion and allocation of design variables

all alternatives defined by the same design variables into architectures further enables to reduce the number of architectures to be optimized to four. Doing so allows for the number of discrete algorithms to be set to be reduced by a factor of $2 \times 10^8$.

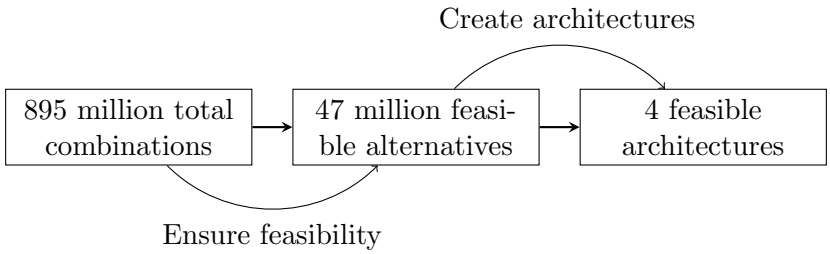



**Figure 117:** Summary of the architecture generation process

The application of the proposed approach on suborbital vehicles shows that all suborbital vehicles can be grouped into four architectures, as described below:

- Architecture 1: slender vehicles that take off vertically without jet engines
- Architecture 2: winged vehicles that take off from the ground horizontally or vertically

without jet engines

- Architecture 3: winged vehicles that take off from the ground with jet engines
- Architecture 4: winged vehicles without jet engines launched from an aircraft and that lands horizontally

### 9.2.4 Step 3: Evaluate Alternatives

All generated alternatives are evaluated using a two-step process. The design framework developed in Chapter 6 is first used to evaluate passenger experience, vehicle safety, and vehicle life-cycle costs. Then, a program framework is used to determine the NPV based on the outputs from the design framework and additional business variables. Figure 118 summarizes the modeling and simulation environment used for this experiment.

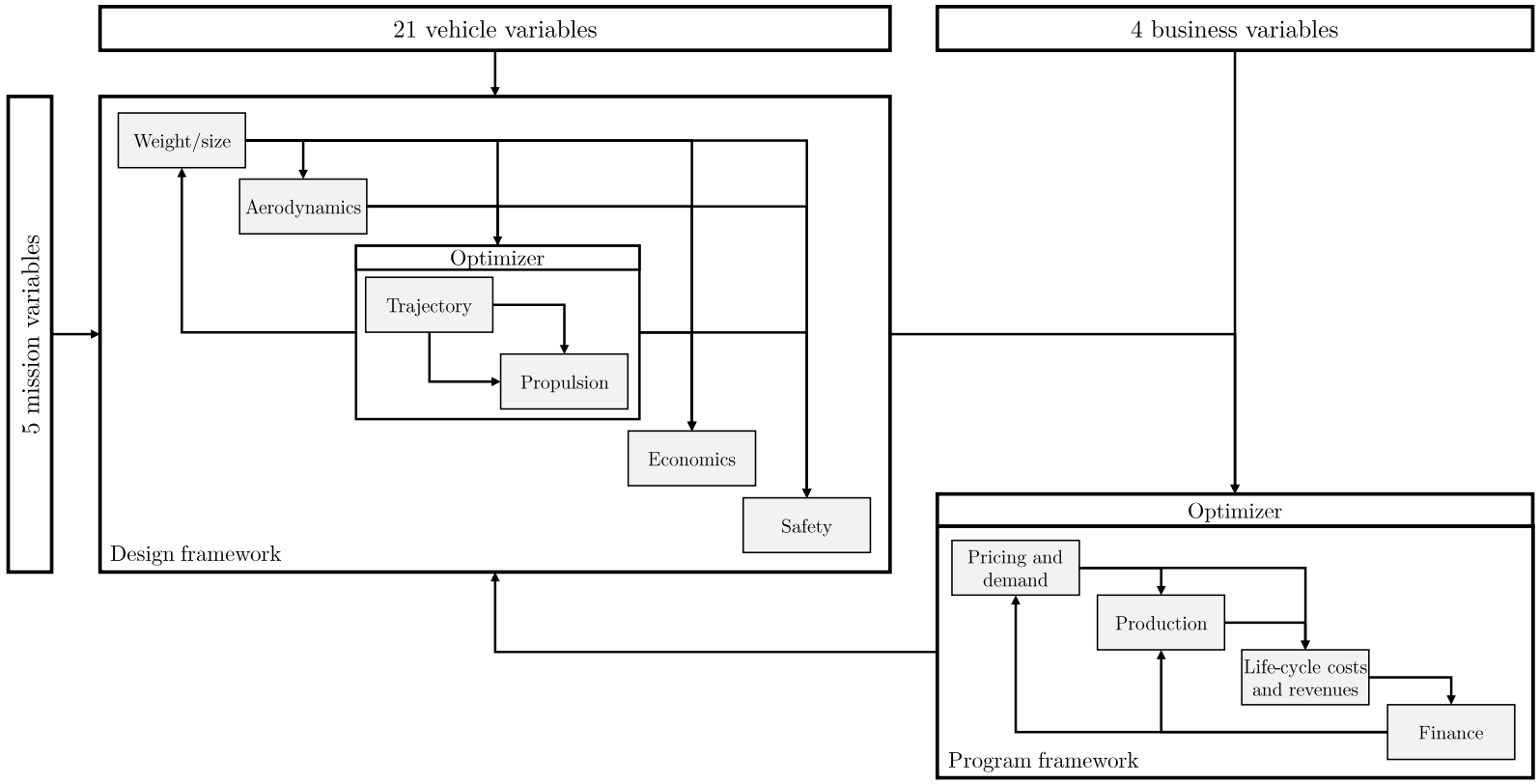



**Figure 118:** Overview of the modeling and simulation environment

#### *9.2.4.1 Design Framework*

As described in Chapter 6, the design framework follows a modified MDF structure and is composed of six disciplinary modules: weight/size, aerodynamics, trajectory, propulsion, economics, and safety. The design framework requires 27 vehicle variables, 7 mission variables, and 4 constraints, as described below:

- Vehicle variables: presence of wing, presence of horizontal tail, presence of vertical tail, presence of jet engines, fuselage base diameter, length of the back part, length of the front part, chamber pressure, nozzle expansion ratio, propellant, thrust of the rocket engine, number of jet engines, thrust of the jet engines, bypass ratio, presence of afterburner, turbine inlet temperature, wing surface area, wing thickness-to-chord ratio, wing sweep angle, wing aspect ratio, wing taper ratio, vertical tail sweep angle, vertical tail aspect ratio, horizontal tail sweep angle, horizontal tail aspect ratio, seat pitch, and fuselage diameter.
- Mission variables: take-off mode, landing mode, transition altitude, number of pilots, maximum altitude, number of passengers, and duration of the program.
- Constraints: maximum load factor, maximum dynamic pressure, maximum temperature, and length of the runway.

The six disciplinary modules are described below:

- Weight/size: determine both the weight and the dimensions of each component of the vehicle.
- Aerodynamics: determine the aerodynamic coefficients of the vehicle.
- Trajectory: determine the best trajectory for a given vehicle and propulsion system.
- Propulsion: determine the propulsion characteristics required to compute the trajectory.
- Economics: determine all life-cycle cost components of the vehicle.

- Safety: determine the risk level of the vehicle.

An intermediate optimizer that groups the trajectory and the propulsion modules is also developed to enforce the design constraints and reduce the number of feedback loops. The design framework is able to characterize the flying, safety, and economic performance of each vehicle. In addition, a program framework is needed to optimize the overall suborbital program with respect to profitability, as described in the next section.

#### *9.2.4.2 Program Framework*

In order to process the business-related and program-related parts of the analysis and optimization, a program framework composed of four modules is developed. In order to optimize the program at a high-level, four business-related variables are selected, as described below:

- Ticket price: the price to pay per person per flight.
- Number of flights per year: the number of flights performed by each vehicle each year in order to best fit the demand. This number is limited to 104 (or 2 flights per week).
- Number of vehicles: the number of vehicles to produce.
- Debt proportion: the relative amount of debt taken by the company compared to the sum of debt and equity taken. While taking on some debt can have advantages, it can become unprofitable as the interest rates become higher.

This section describes the model used for each discipline: demand forecast, production, life-cycle costs and revenues, and financial analysis.

**Pricing and Demand Forecast**

As presented in Section 1.1.1.1, the demand for commercial suborbital flights is highly sensitive to ticket price. In order to find the optimum design, it is important to quantify the potential demand and its sensitivity. For that purpose, many studies have already been conducted to assess the short and medium-term suborbital market for space tourism [31, 86,

163, 188, 250, 435]. Results from surveys conducted by Airbus Group (previously EADS), Virgin Galactic, and the Futron Corporation are used, as shown in Figure 119.

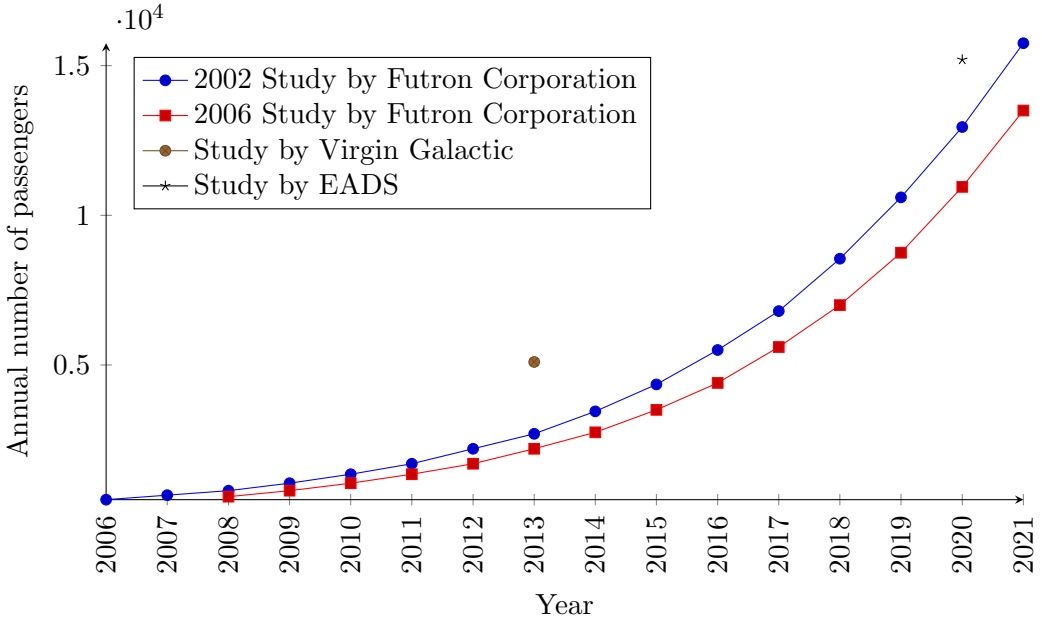



**Figure 119:** Potential market growth for suborbital flights [31, 163, 250]

While these data are outdated, it is assumed that the demand will follow the same trend once the first vehicle has been certified. This forecast assumes a ticket price that drops from \$100,000 to \$50,000 per flight and per passenger during the first five years. Based on these data and the price sensitivity identified by the Tauri Group [435], Equation 165 provides the model of the annual demand $N_d$ as a function of the ticket price $t_p$ in 2015 U.S. k\$ used in the proposed scenario.

$$N_d\left(t_p\right) = 30.894\left[739\exp\left(-0.01149t_p\right) + 74.1\exp\left(-0.001286t_p\right)\right] \tag{165}$$

Figure 120 shows the developed model for the price sensitivity of the demand for the commercial suborbital market.

The revenues that can potentially be generated by the commercial suborbital market are displayed in Figure 121. Ideally, around \$930 million can be generated annually with 8,500 passengers for a ticket price at around \$110k. Figure 121 also shows that two attractive segments can be identified. The first segment corresponds to high-frequency and relatively

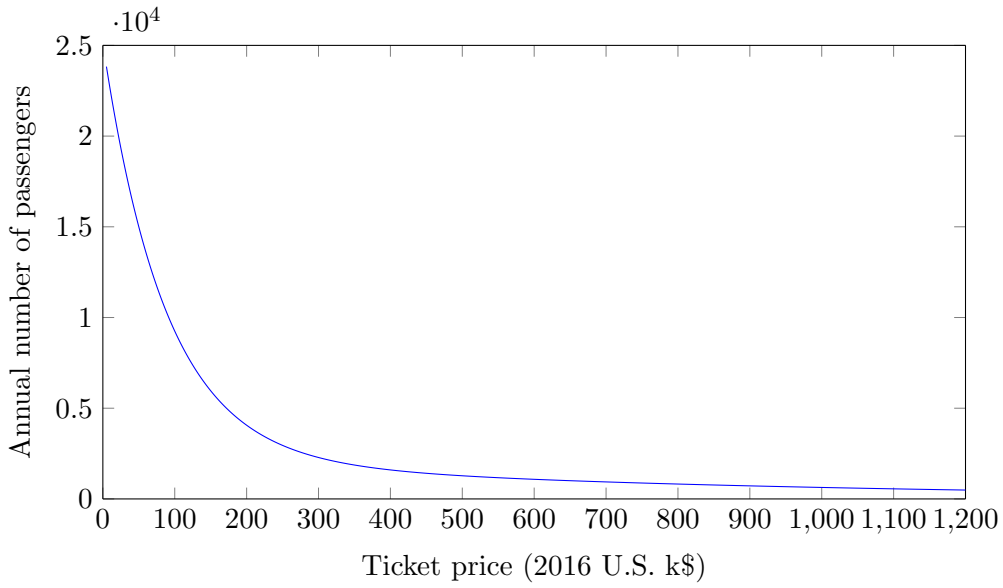



**Figure 120:** Price sensitivity of the demand for the commercial suborbital market

affordable flights (around $110k). Assuming a capacity of four to five passengers per vehicle, this segment corresponds to about 1,850 flights per year (or five flights per day performed by all available vehicles).

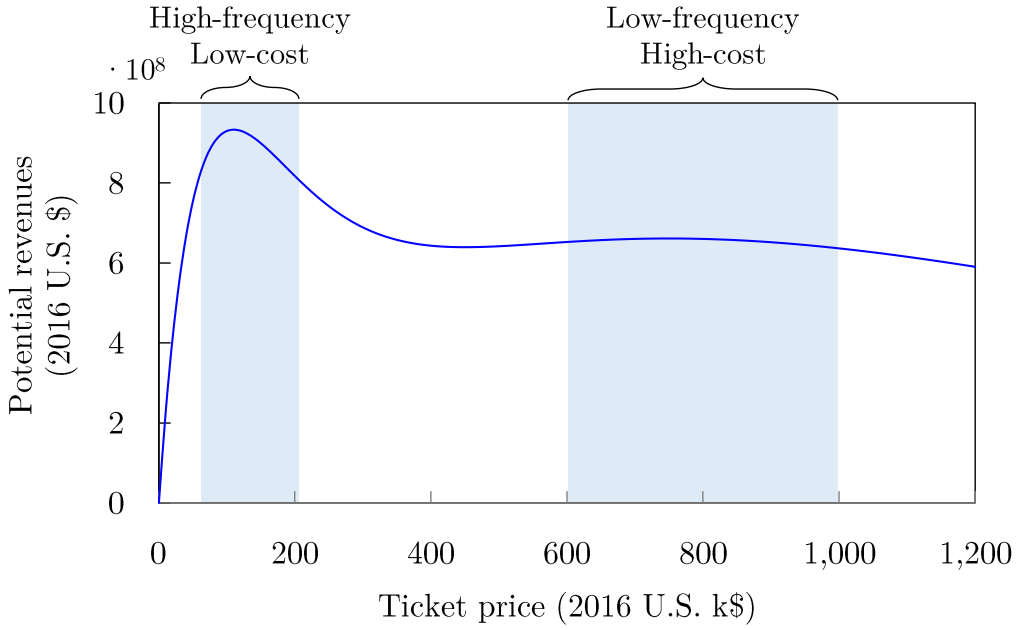



**Figure 121:** Potential revenues generated by the commercial suborbital market

The second segment corresponds to low-frequency and expensive flights (around $750k). Based on the same assumptions, this segment only requires 200 flights per year (or one flight

every two days). The potential revenues generated by 880 passengers can reach around $660 million. Based on this analysis, two major strategies tend to emerge:

- Affordable and high-frequency flights that aim at democratizing the suborbital market.
- Expensive and low-frequency flights that aim at keeping the suborbital market for wealthy customers.

**Production and Capacity**

The production model implemented in this experiment relies on a simplified approach. Given a requested number of vehicles to produce, the company starts producing them sequentially from Year 6 (after 5 years needed for R&D), while respecting a constrained maximum production rate per year (5 vehicles/year).

As production dictates capacity, the overall capacity can then be computed using Equation 166 that states that the demand $N_{PAX,k}$ the company can fulfill at year $k$ is the product of the passenger capacity of the aircraft $n_{PAX}$, the number of flights per vehicle per year $n_{flights}$, and the number of suborbital vehicles $n_{vehicles,k}$ available at year $k$.

$$N_{PAX,k} = n_{PAX}.n_{flights}.n_{vehicles,k} \tag{166}$$

At the end of the program cycle, at Year 17, the number of vehicles available is assumed to decrease by a quarter of the maximum capacity, as vehicles start being retired.

**Life-Cycle Costs and Revenues**

Once demand, production, and capacity are known, it is possible to determine the time distribution of the costs and revenues during the life-cycle of the program using the distribution of all life-cycle cost components outputted by the design framework, as described below:

- **Costs:** Costs stem from four different categories: RDT&E, production, operations, and carrier aircraft.

– *RDT&E costs:* the total amount of RDT&E costs is directly given by the design framework. By emulation from ALCCA [307], the development period is assumed to be six years, with the cost distribution being 5%, 20%, 20%, 20%, 20%, and 15%, as displayed in Figure 122.

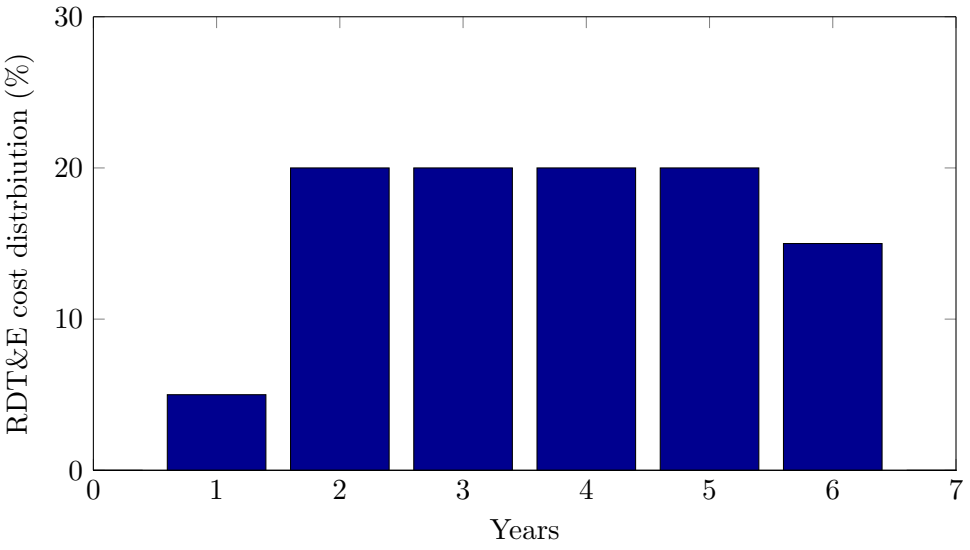



**Figure 122:** Distribution model of RDT&E costs

– *Production costs:* production costs model the cost of manufacturing the different vehicles that are created during the program life-cycle. The design framework provides the production cost per unit assuming that 20 vehicles are produced in order to ensure that the maximum demand can be satisfied with a one-passenger vehicle. In order to account for the learning curve effect, the unit cost is corrected, by moving up or down along the learning curve. Once the unit cost is known, the distribution of costs from ALCCA is replicated, by setting the cost of producing each aircraft to 40% of the unit cost the year preceding production, and 60% the year the vehicle is produced.

– *Operating costs:* operating costs represent all the costs needed to operate the vehicles during the operation phase of the life-cycle. While the operating cost components per flight are provided by the design framework, it assumes a fleet of 20 vehicles flying a given number of flights per vehicle per year. These costs

are corrected, and the total operating costs per year are calculated.

  - *Carrier aircraft costs:* carrier aircraft costs represent a non-recurring expense related to the purchase of an aircraft that will carry the vehicle to a sufficient altitude before releasing it.

- **Revenues:** revenues directly depend on demand $N_d$ and capacity $N_{PAX}$, as well as the ticket price per passenger $t_p$. For each year $k$, revenues $R_k$ are determined using Equation 167.

$$R_k = \min(N_{PAX,k}, N_{d,k}) \times t_p \tag{167}$$

**Financial Analysis**

The goal of the financial analysis is to compute the NPV, as well as some other financial metrics of interest.

As discussed in Chapter 4, the NPV can be computed using Equation 168, where $N$ is the number of periods, $FCF_k$ the cash flow at each period $k$, and $i$ the discount rate.

$$NPV = \sum_{k=1}^{N} \frac{FCF_k}{(1+i)^k} \tag{168}$$

$FCF_k$ can be computed from the life-cycle costs and revenues obtained in the previous module. Indeed, the general equation for $FCF$ is given in Equation 169, where $EBITDA$ is the Earnings Before Interests, Taxes, Depreciation, and Amortization (EBITDA) (which can be assumed to be revenues minus operating costs), $C_{tax}$ the amount of taxes paid, $\Delta NWC$ the change in Net Working Capital (NWC), which is assumed to be equal to zero, and $Capex$ the capital expenditure, which is the production costs plus the RDT&E costs.

$$FCF = EBITDA - C_{tax} - \Delta NWC - Capex \tag{169}$$

As a result, $FCF_k$ can be expressed as a function of revenues at period $k$ ($R_k$), and total costs at period $k$ ($C_k$), as shown in Equation 170. As the free cash flow equation does

not include interest rates, depreciation, and amortization, they should not be included in the total costs.

$$FCF_k = R_k - C_k \tag{170}$$

As commonly done by investors, the discount rate $i$ is assumed to be equal to the Weighted Average Cost of Capital (WACC). The latter represents the cost of a firm's capital where each category of capital is proportionately weighted (common stock, preferred stock, bonds, long-term debt, etc.). It can be calculated using Equation 171, where $E$ is the equity, $D$ the debt, $r_e$ the cost of equity, $r_d$ the cost of debt, and $\tau$ the corporate tax rate.

$$WACC = \frac{E}{D+E} r_e + \frac{D}{D+E} r_d (1-\tau) \tag{171}$$

While the official corporate tax rate is fixed at 35% in the U.S., the actual corporate tax rate paid by the company can be different. For example, a company making zero profits before interests and taxes on a specific period will not pay any tax for this period, and therefore will not be able to deduct any debt interest from its taxes. Companies also have various ways and tools to decrease the amount of tax they pay. The calculation of the WACC also involves the costs of debt and equity.

The cost of equity $r_e$ is commonly determined using the Capital Asset Pricing Method (CAPM) introduced by Sharpe [394] and Lintner [260]. The coefficient "beta" $\beta$ of the company has to be known initially (measure of the volatility, or systematic risk, of a security or a portfolio in comparison to the market as a whole), as well as the market risk premium $\mathrm{E}[r_m - r_f]$, which represents the surplus of return that investors are expecting from the market compared to U.S. treasury bonds, due to its riskiness. The CAPM method is presented in Equation 172.

$$r_e = r_f + \beta \, \mathrm{E}[r_m - r_f] \tag{172}$$

If the beta coefficient is not known, it is possible to find the levered beta $\beta_L$ using the

average unlevered beta $\beta_U$ of equivalent companies using Equation 173. The unlevered beta, or asset beta, is the beta of a company assuming zero debt, while the levered beta, or equity beta, includes financial leverage as there is a positive correlation between beta and the amount of debt a company has in its initial financial structure [102]. This method also enables trade studies to find the optimal capital structure of a company.

$$\beta_L = \beta_U \left(1 + (1 - \tau)\frac{D}{E}\right) \tag{173}$$

In this particular problem, as most suborbital companies are private, it is hard to find comparable companies' betas. For that reason, an unlevered beta of 1.1 is assumed, which reflects a higher risk for these companies, even before being indebted.

In order to determine the cost of debt $r_d$, the ratings given to the company by major credit rating agencies such as Standard and Poor's, Moody's, and Fitch Group are used [299]. Based on these ratings, it is then possible to determine the expected yield spread for the corporate bonds, as displayed in Table 51. Fairly intuitively, the better the credit rating, the lower the yield spread. This yield spread $\Delta r$ is then added to the yield of the U.S. treasury bonds' yield, often considered as a representation of the risk-free rate $r_f$, as displayed in Equation 174 [129, 147]. As the debt is assumed to be issued for the whole duration of the project, the most recent estimates of the default spreads are used.

$$r_d = r_f + \Delta r \tag{174}$$

**Table 51:** Yield spreads based on credit ratings on November 2015 [147]

| AAA | AA | A | BBB | BB | B |
|---|---|---|---|---|---|
| 0.76% | 0.95% | 1.20% | 2.18% | 3.88% | 6.20% |

While this method is sufficient to compute $r_d$ for a rated company, it does not account for the variations of credit rating when a company significantly changes its debt level, and does not enable to compute the yield spread for a notional company without official ratings. In this experiment, the company does not have yet a rating. Thus, an additional analysis

has to be made.

To overcome this problem, the notion of Interest Coverage Ratio (ICR) is introduced. The ICR can be computed as the ratio of the Earnings Before Interests and Taxes (EBIT) of a company over the amount of interests it pays, as shown in Equation 175.

$$ICR = \frac{EBIT}{Interests} \tag{175}$$

In other terms, the ICR shows the ability of a firm to pay for its interests at every period. The ICR can be directly related to credit rating and default spread. Indeed, the easier for them it is to pay, the more confident creditors are that the company will not default. Table 52 shows the relation between ICR, credit rating, and default spreads [102].

**Table 52:** Credit rating and default spread in January 2016 [102]

| Interest coverage ratio is | | Rating | Spread |
|---|---|---|---|
| Greater than | $\leq$ to | | |
| 12.5 | 100000 | Aaa/AAA | 0.75% |
| 9.5 | 12.499999 | Aa2/AA | 1.00% |
| 7.5 | 9.499999 | A1/A+ | 1.10% |
| 6 | 7.499999 | A2/A | 1.25% |
| 4.5 | 5.999999 | A3/A- | 1.75% |
| 4 | 4.499999 | Baa2/BBB | 2.25% |
| 3.5 | 3.9999999 | Ba1/BB+ | 3.25% |
| 3 | 3.499999 | Ba2/BB | 4.25% |
| 2.5 | 2.999999 | B1/B+ | 5.50% |
| 2 | 2.499999 | B2/B | 6.50% |
| 1.5 | 1.999999 | B3/B- | 7.50% |
| 1.25 | 1.499999 | Caa/CCC | 9.00% |
| 0.8 | 1.249999 | Ca2/CC | 12.00% |
| 0.5 | 0.799999 | C2/C | 16.00% |
| -100000 | 0.499999 | D2/D | 20.00% |

To simulate a suborbital vehicle company, it is assumed that creditors will not have great confidence in the chances of success. Therefore, they would not grant more than a BBB rating, which still belongs to the investment grade category.

One can note that, in this implementation, a series of linear functions is used to ensure

continuity, rather than the suggested step functions. Moreover, an iterative calculation must be adopted. Indeed, the ICR depends on the amount of interests paid. The amount of interests paid depends on the default spread, itself depending on the ICR. Therefore, the computation of ICR and default spread is circularly run until convergence and the results are presented in Figure 123 for both average companies (with a market capitalization smaller than $5 billion) and a suborbital company.

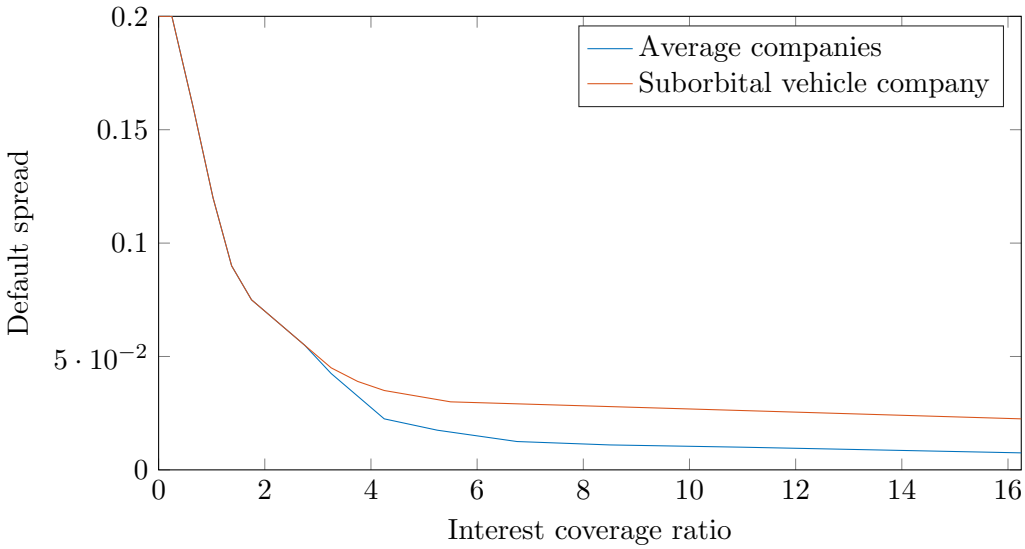



**Figure 123:** Model of the default spread as a function of the interest coverage ratio

The integration of all the aforementioned modules into a single modeling and simulation environment allows designers to estimate the flying, economic, and safety performance of all suborbital vehicles. This information will be used by the decision-making environment to support decision makers in the development of future suborbital programs, as discussed in the next section.

### 9.2.5 Step 4: Make Informed Decisions

Each piece of the decision-making process has been developed and discussed in Chapter 7. First, based on the architecture definition given by the design space definition step, all non-dominated solutions are determined using the evolutionary multi-architecture multi-objective optimization algorithm developed in Section 7.2. Then, requirement uncertainty is infused in the process using fuzzy set theory. This technique allows designers to model

each uncertainty source by a mean value and a standard deviation. The last step of the analysis consists in adding time dependence to the standard deviation of each uncertainty source. All these optimization steps feed into a decision module that provides traceable and quantitative information to decision makers. The information presented provides the ability to conduct trade-offs, identify trends, and prioritize objectives using MADM techniques. Finally, a dynamic and parametric visualization environment is developed to help designers get an overview of the selected concept(s). The overall decision-making process is presented in Figure 124. The implementation of each of these elements is provided in this section.

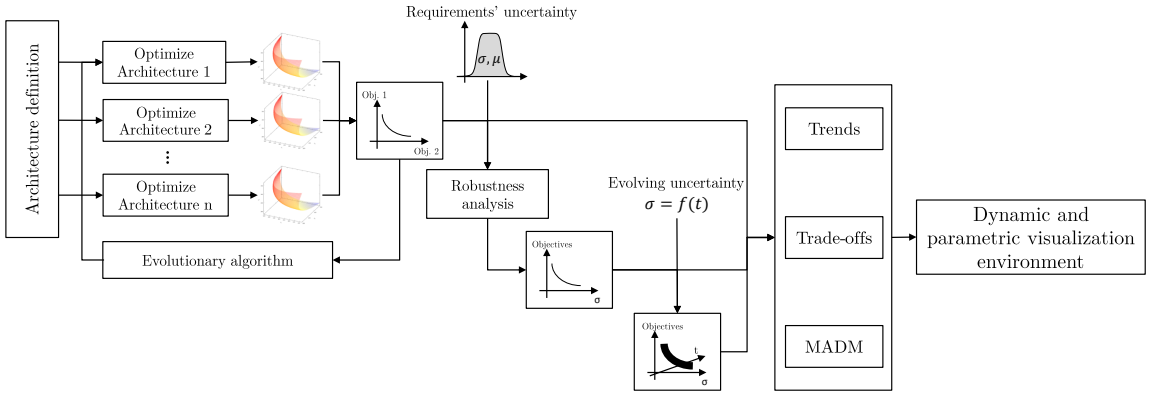



**Figure 124:** Overview of the decision-making process

### *9.2.5.1 Evolutionary Multi-Architecture Multi-Objective Optimization Algorithm*

The optimization is based on 30 variables: 21 vehicle variables, 5 mission variables, and 4 business variables. Table 53 presents all variables used in the optimization along with their ranges and corresponding architectures. As discussed in Section 6.5, three main constraints are added to the optimization: maximum dynamic pressure, maximum load factor, and maximum temperature.

Two series of optimization have been executed in order to produce both two-dimensional and three-dimensional Pareto frontiers. The configuration parameters specific to each series are described in Table 54.

**Table 53:** Definition of the design variables used in the optimization

| | Variables | Ranges | Architectures | | | |
|---|---|---|---|---|---|---|
| | | | **1** | **2** | **3** | **4** |
| **Vehicle** | Propellant | 1, 2, 3, 4, 5, 6, 7, 8 | ✓ | ✓ | ✓ | ✓ |
| | Nozzle expansion ratio | $[2; 100]$ | ✓ | ✓ | ✓ | ✓ |
| | Rocket engine chamber pressure (MPa) | $[2; 12]$ | ✓ | ✓ | ✓ | ✓ |
| | Rocket engine thrust (kN) | $[50; 700]$ | ✓ | ✓ | ✓ | ✓ |
| | Number of jet engines | 1, 2, 3, 4 | | | ✓ | |
| | Jet engine thrust (kN) | $[5; 100]$ | | | ✓ | |
| | Bypass ratio | $[0; 1]$ | | | ✓ | |
| | Afterburner | Yes, No | | | ✓ | |
| | Turbine inlet temperature (K) | $[1,500; 2,500]$ | | | ✓ | |
| | Diameter of the fuselage base (m) | $[0; 1]$ | ✓ | ✓ | ✓ | ✓ |
| | Length of the aft fuselage (m) | $[0.1; 1]$ | ✓ | ✓ | ✓ | ✓ |
| | Length of the front fuselage (m) | $[1; 5]$ | ✓ | ✓ | ✓ | ✓ |
| | Wing surface ($m^2$) | $[10; 100]$ | | ✓ | ✓ | ✓ |
| | Wing thickness-to-chord ratio | $[0.08; 0.14]$ | | ✓ | ✓ | ✓ |
| | Wing sweep angle (rad) | $[0; 1.4]$ | | ✓ | ✓ | ✓ |
| | Wing aspect ratio | $[1; 6]$ | | ✓ | ✓ | ✓ |
| | Wing taper ratio | $[0; 1]$ | | ✓ | ✓ | ✓ |
| | Vertical tail aspect ratio | $[1; 6]$ | | ✓ | ✓ | ✓ |
| | Vertical tail sweep angle (rad) | $[0; 1]$ | | ✓ | ✓ | ✓ |
| | Horizontal tail aspect ratio | $[1; 6]$ | | ✓ | ✓ | ✓ |
| | Horizontal tail sweep angle (rad) | $[0; 1]$ | | ✓ | ✓ | ✓ |
| **Mission** | Take-off mode | 1, 2 | | ✓ | | |
| | Landing mode | 0, 1 | | | ✓ | |
| | Transition altitude (km) | $[5; 18]$ | | | ✓ | ✓ |
| | Number of pilots | 0, 1, 2 | ✓ | ✓ | ✓ | ✓ |
| | Number of passengers | 1, 2, 3, 4, 5, 6, 7, 8 | ✓ | ✓ | ✓ | ✓ |
| **Business** | Debt proportion (%) | $[0; 100]$ | ✓ | ✓ | ✓ | ✓ |
| | Number of vehicles | $[0; 20]$ | ✓ | ✓ | ✓ | ✓ |
| | Ticket price (2016 U.S. k$) | $[20; 1,500]$ | ✓ | ✓ | ✓ | ✓ |
| | Flights per month per vehicle | $[1; 10]$ | ✓ | ✓ | ✓ | ✓ |

#### *9.2.5.2 Uncertainty Propagation*

As discussed in Section 7.1, fuzzy set theory is used to propagate uncertainty through the design and program frameworks.

The first step consists in using the uncertainty parameters defined in Step 1 to compute

**Table 54:** Setup parameters of the optimization algorithm

| Parameters | 2D optimization | 3D optimization |
|---|---|---|
| **Objectives** | NPV, safety | NPV, safety, passenger experience |
| **Population** | 100 | 500 |
| **Maximum generations** | 60 | 100 |

the standard deviation of the NPV, which is used as another objective for the optimization problem. The first derivative required to calculate the gradient is determined using the Newton's difference quotient, as suggested in Section 7.1, with a calculation-step $h$ equal to 5%. Based on these data, the probability of having a positive NPV is also evaluated as it can be used as a metric for program risk.

The second step consists in scaling each uncertainty source using time-based scaling factors. Hence, time becomes another dimension of the optimization.

#### *9.2.5.3 Decision Platform*

The decision platform is composed of various types of plots that support multi-objective trade-off analyses and trends identification:

- Pareto frontiers: two and three-dimensional plots that display the Pareto frontier of solutions are created to highlight trade-offs between objectives. In addition, grouping solutions with respect to different parameters such as architecture, propellant mixture, and number of passengers supports the identification of key trends.
- Ternary plot: ternary plots facilitate the identification of dominant concepts.
- Multi-Attribute Decision Making: the probabilistic TOPSIS developed in Section 7.3 is applied to various scenarios.
- Parametric and dynamic visualization dashboard: the concepts of interest can be visualized using the dashboard developed in Section 7.4, which links the results of the selection process to the CAD software Catia.
- Concept description: once concepts of interest have been identified, a multi-disciplinary

description is provided to quantify their performance and facilitate discussions.

The results generated by the proposed experiment are provided in the next section. The performance of the proposed methodology will then be assessed using the validation criteria identified in Section 9.1.

## *9.3 Results*

In order to validate the proposed methodology and the Overarching Hypothesis, the validation criteria identified in Section 9.1 are evaluated using the results provided by the overall experiment.

### 9.3.1 Methodology Flexibility

To assess the first criterion, the flexibility of the methodology is evaluated. In particular, one must determine if alternatives evaluated using different design frameworks can be systematically compared and optimized. From an optimization process' perspective, a design framework is exhaustively defined by four key elements:

- A set of inputs: each design framework requires a specific set of inputs. Each input is characterized by a variable along with its specific range, within which the design framework is valid.
- A set of outputs: the objective of the design framework is to perform an analysis in order to provide a set of outputs to the users.
- A set of constraints: the design space that can be handled by a design framework might be limited using a set of constraints.
- A black box: the models embedded are represented by a black box that uses both the set of inputs and the set of constraints to generate the desired outputs.

By definition and for consistency purposes, if alternatives are compared, they must be evaluated against the same metrics. Hence, all design frameworks must have the same outputs. In addition, the variable-oriented morphological analysis allows designers to systematically

generate variable-oriented architectures that will feed into the proposed evolutionary multi-architecture multi-objective optimization algorithm. This enables each architecture, characterized by a specific set of variables and evaluated with a specific design framework, to be embedded into a single optimization process to be systematically compared and optimized. As a consequence, by construction, the proposed optimization algorithm can handle multiple design frameworks and systematically compare and optimize all alternatives. This leads to the validation of the first criterion:

**Validation Criterion 1: Alternatives evaluated with different modeling and simulation environments can be systematically compared and optimized.**

### 9.3.2 Computational Efficiency of the Methodology

The validation of Criterion 2 requires the evaluation of the computational time required for exploring the design space with both the proposed methodology and a full factorial DoE of all alternatives identified during the design space definition. Indeed, those two methodologies are the only ones capable of exhaustively covering the design space.

As detailed in Figure 115, there are 895,795,200 possible solutions. The evaluation of each solution takes around four seconds if no design variables are infused. As a consequence, using a full factorial DoE, the evaluation of all possible combinations would take around 3,583,180,800 seconds, or 114 years. If the proposed methodology is partially used to remove all unfeasible combinations, the evaluation would still take around six years.

The time required to explore the design space with the proposed methodology depends on two parameters selected by designers: maximum number of generations $N_g$ and population (number of individuals) of each stochastic optimization algorithm $N_p$. Equation 176 provides the maximum evaluation time $T^*$ required for the proposed methodology, where $t_{se}$ is the time required to evaluate a single concept, and $N_a$ the number of architectures.

$$T^* = N_g N_a N_p t_{se} \tag{176}$$

While the infusion of design variables increases the evaluation time of a single alternative from four seconds to twelve seconds, it also allows designers to consider an infinite number

of solutions. Since four architectures have been identified for suborbital vehicles, Table 55 shows the computational time required for multiple combinations of population and maximum number of generations. This assumes that all architectures have been optimized with the maximum number of generations.

**Table 55:** Required execution time of the proposed methodology (days)

| | | **Maximum number of generations** | | | |
|---|---|---|---|---|---|
| | | **100** | **200** | **500** | **1,000** |
| **Population** | **100** | 5 | 11 | 28 | 56 |
| | **200** | | 22 | (56) | 111 |
| | **500** | | | 139 | 278 |
| | **1,000** | | | | 555 |

Based on the guidelines provided by German [171], one can assume that 200 individuals are enough to get accurate results. In addition, a maximum number of generations equal to 500 is also suggested by Michalewicz et al. [183, 295]. Using these guidelines, the evaluation of the four architectures requires around 56 days. Hence, the proposed methodology highly decreases the computational time while also increasing the number of concepts considered. This leads to the validation of the second criterion:

**Validation Criterion 2: The proposed methodology decreases the computational time by a factor of 750 while also increasing the number of concepts investigated.**

### 9.3.3 Design Space Coverage Capabilities

In order to validate Criterion 3, the capabilities of the proposed methodology are compared to two other approaches: the DoE and the single-architecture optimization. The first approach simulates a high-level qualitative exploration of the design space and the second one an optimization based on a baseline selected by experts.

#### *9.3.3.1 Comparison with a Design of Experiments*

One of the most popular techniques for design space exploration is the DoE. This technique consists in optimally covering the design space by generating points, which follow a specific patterns. Latin Hypercube DoEs are usually used as they provide a good coverage of the interior of the design space. In order to compare the performance of such methodology with the proposed methodology, a DoE of 6,000 points is individually performed for each architecture. This corresponds to the maximum number of points that can be reached by the proposed algorithm. While surrogate models are usually created to improve the performance of DoEs, it is not suitable for complex problems. Since there are discrete and categorical variables, one set of surrogate models would have to be created and validated for each combination of categorical/discrete variables. As a consequence, the results of the DoE are directly compared to the ones obtained with the proposed algorithm. In particular, the Pareto frontiers of points optimized with respect to both NPV and safety using both techniques are displayed in Figure 125.

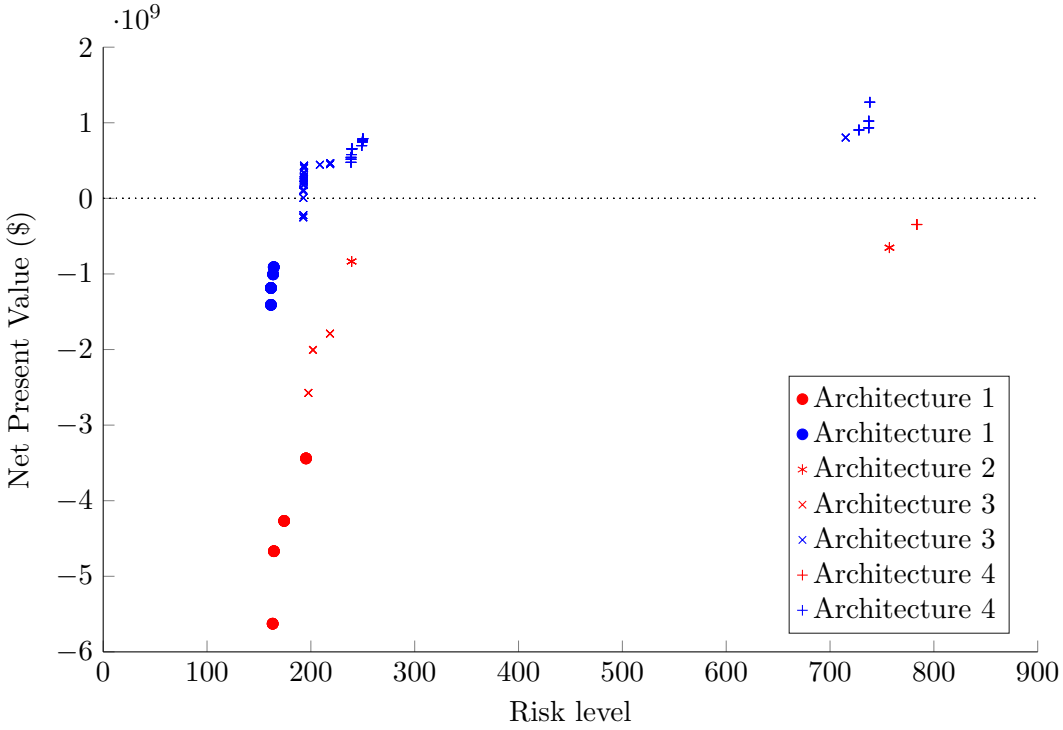



**Figure 125:** Comparison between the proposed methodology (blue) and the DoE (red)

As shown in Figure 125, the Pareto frontier generated by the proposed algorithm (blue) provides better results. Indeed, none of the points generated by the DoE is non-dominated. In addition, the average distance of the points on the Pareto frontier to the ideal solution is 39% smaller for the proposed algorithm. The Pareto frontier generated by the proposed methodology also has more that four times as many points as the one generated by the DoE (44 against 10).

Focusing individually on each objective, it can be noticed that the application of the proposed algorithm provides an improvement of $1.6 billion in NPV and 1.7 in risk level. In particular, the DoE does not present any profitable concept, while the proposed algorithm highlights a concept that can make up to $1.3 billion in NPV. This shows the additional value that can be captured by the proposed methodology when compared to a DoE.

#### *9.3.3.2 Comparison with a Single Architecture Optimization*

In this section, the Pareto frontier of solutions obtained with the proposed algorithm is compared to the one obtained using a single-architecture optimization methodology. This approach consists in finding the best solution(s) for a given architecture that has been defined by experts or based on high-level qualitative analyses. This approach is similar to the results that can be obtained using the architecture optimization approach if two or three architectures are selected. Figure 126 displays the Pareto frontiers obtained for each architecture ($\cdot$) along with the one obtained with the proposed methodology and colored by architecture ($\circ$).

As shown in Figure 126, none of the architecture results in a coverage as accurate and extensive as the one provided by proposed methodology. Each architecture populates a given region of the design space: Architecture 1 is the safest, Architecture 4 is the most profitable, and Architecture 3 is an intermediate solution. Due to the complexity of the problem, such information is not available at the beginning of the design process. Hence, selecting an architecture based on experts' judgment would have resulted in missing some portions of the design space. For instance, if designers had selected Architecture 1, no profitable solution would have been found. To quantify the differences between the two

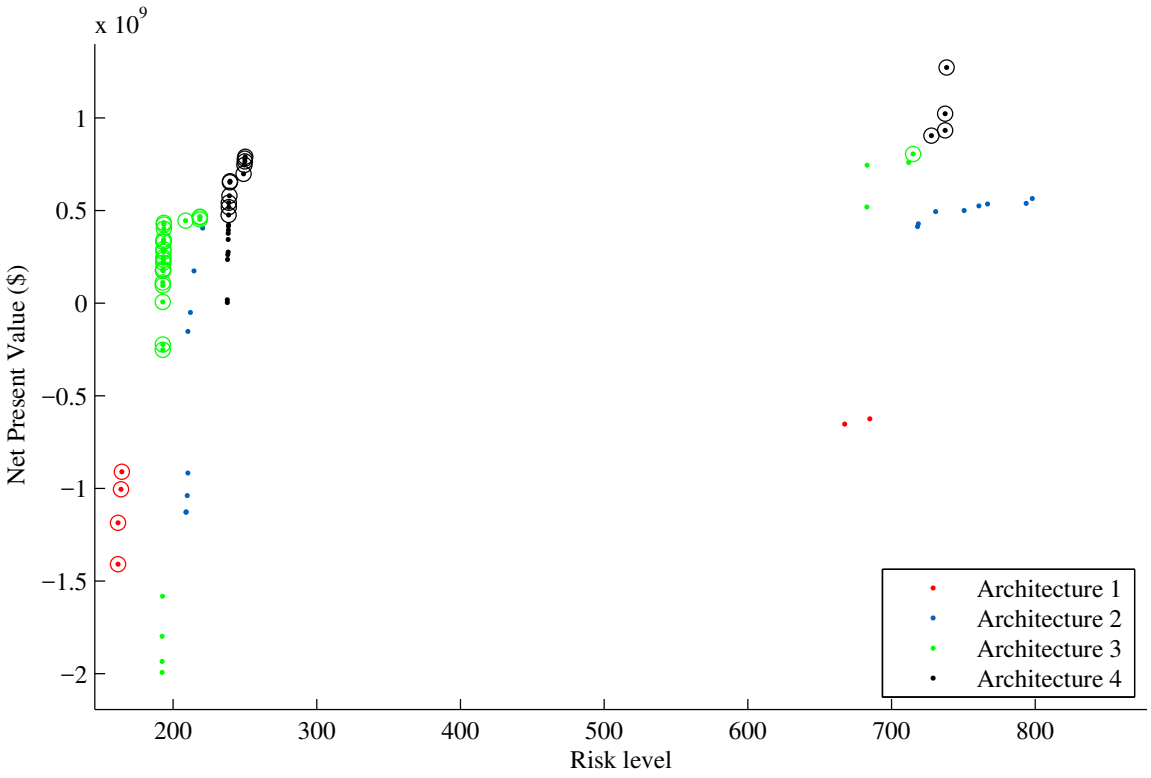



**Figure 126:** Proposed methodology (○) vs. single-architecture optimizations (·)

approaches, the metrics identified in Section 9.1.1 are calculated and compared in Table 56.

**Table 56:** Proposed algorithm versus single-architecture optimization

| Parameters | Single-architecture optimization applied to ... | | | | Proposed methodology |
|---|---|---|---|---|---|
| | Architecture 1 | Architecture 2 | Architecture 3 | Architecture 4 | |
| Best NPV ($ billion) | -0.6 | 0.6 | 0.8 | 1.3 | 1.3 |
| Best risk level | 162 | 209 | 192 | 238 | 162 |
| Number of points | 7 | 16 | 31 | 25 | 44 |
| Average distance | 1.10 | 1.00 | 0.55 | 0.54 | 0.31 |

As shown in Table 56, the proposed methodology, which considers all architectures,

outperforms all single-architecture optimizations. Indeed, the number of points on the final Pareto frontier is larger for the proposed methodology. Similarly, the average distance to the ideal solution is smaller. On average, selecting a single architecture for the comparison would decrease the potential program NPV by 140% and the potential safety level of the vehicle by 20%. Hence, this confirms that the proposed methodology highly improves the coverage of the design space.

As shown in this section, the proposed methodology provides a better coverage of the design space compared to other available approaches, as demonstrated by the comparison of the different metrics: extremum values, average distance, and number of points. This leads to the validation of the third criterion:

**Validation Criterion 3: The proposed methodology provides a better coverage of large design spaces than current approaches.**

### 9.3.4 Ability to Perform Trade-Offs and Identify Trends

When dealing with complex vehicles and multiple objectives, it is extremely important for decision makers to be able to perform quantitative trade-off analyses. This capability allows them to make traceable and informed decisions about which technologies to further investigate and where to focus their research.

To evaluate those capabilities, this section focuses on the observations that can be made using the results provided by the proposed methodology. First, a series of two-dimensional Pareto frontiers are presented. Then, a series of Ternary plots are provided. Finally, dominant concepts are identified through a probabilistic multi-attribute plot.

#### *9.3.4.1 Two-Dimensional Trade-Offs*

Figure 127 provides the final set of optimal vehicles identified from the Pareto frontier of each architecture. The vehicles are colored by number of passengers and marked by architecture. Figure 127 shows the diversity of possible configurations in the final set of points, and illustrates the lack of clear trends that would facilitate any rapid decisions. One can also note that not all vehicles are economically viable. In particular, among the 3,200 vehicles, only 37% provide a positive NPV, and 3% provide an NPV greater than $1 billion.

These observations demonstrate the importance of providing decision makers with the ability to perform quantitative trade-offs and to rigorously select the best concept(s) with respect to their requirements. In order to identify various trends and make new observations, the vehicles are categorized based on their important features: type of propellant, architecture, and number of passengers. The results are presented in Figures 128 to 133.

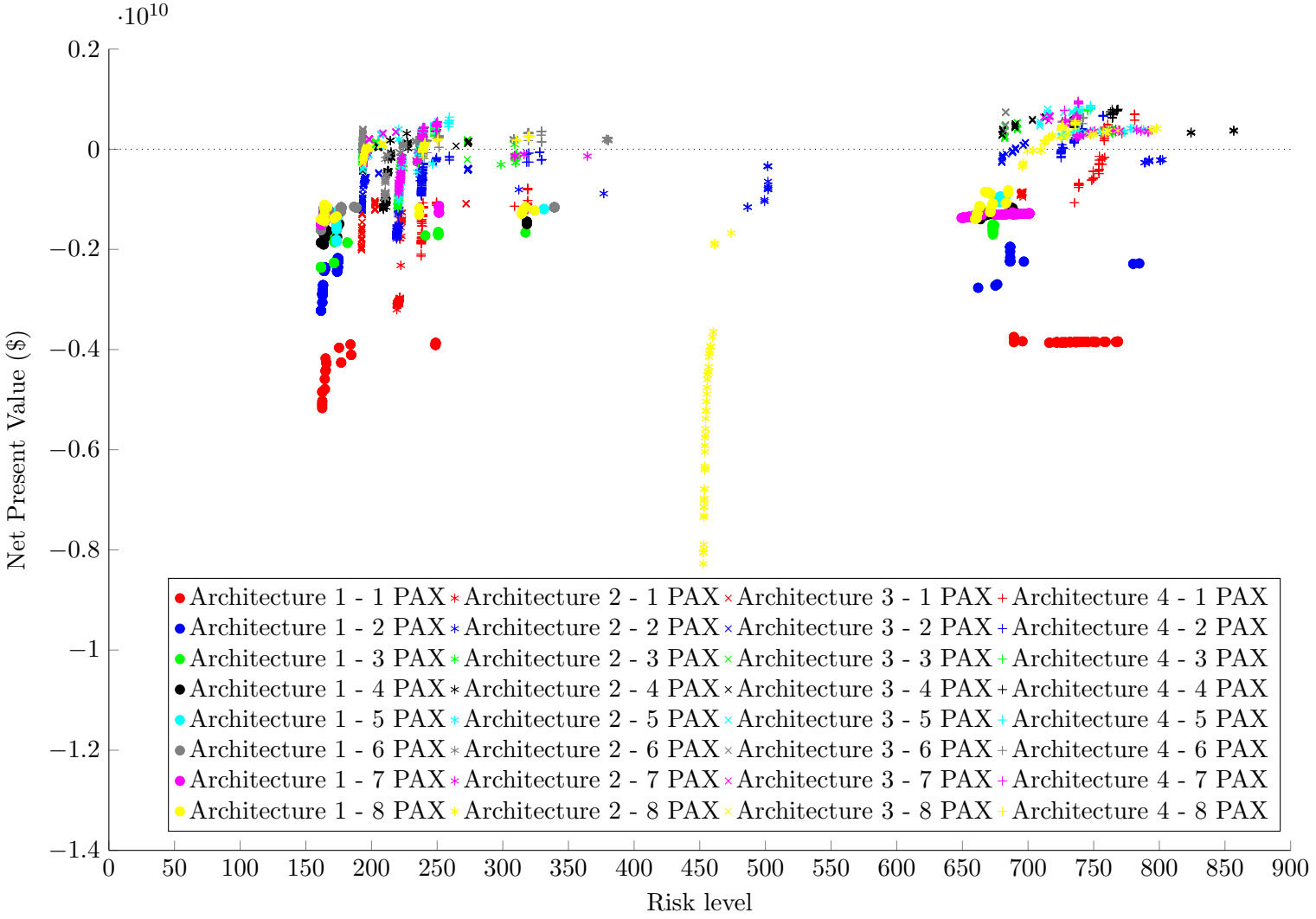



**Figure 127:** Combination of Pareto frontiers originating from the different architectures

Figure 128 provides the combination of all Pareto frontiers originating from the different architectures colored by mixture of propellants. There are three main clusters, which correspond to the type of propellant: solid, liquid, and hybrid. As expected, the most profitable vehicles are powered by solid engines. However, these vehicles are also extremely risky. The safest vehicles are powered by hybrid engines and some of them appear to be profitable. It can also be noticed that none of the vehicles powered by liquid engines are profitable, while all other propellant types are able to power profitable vehicles.

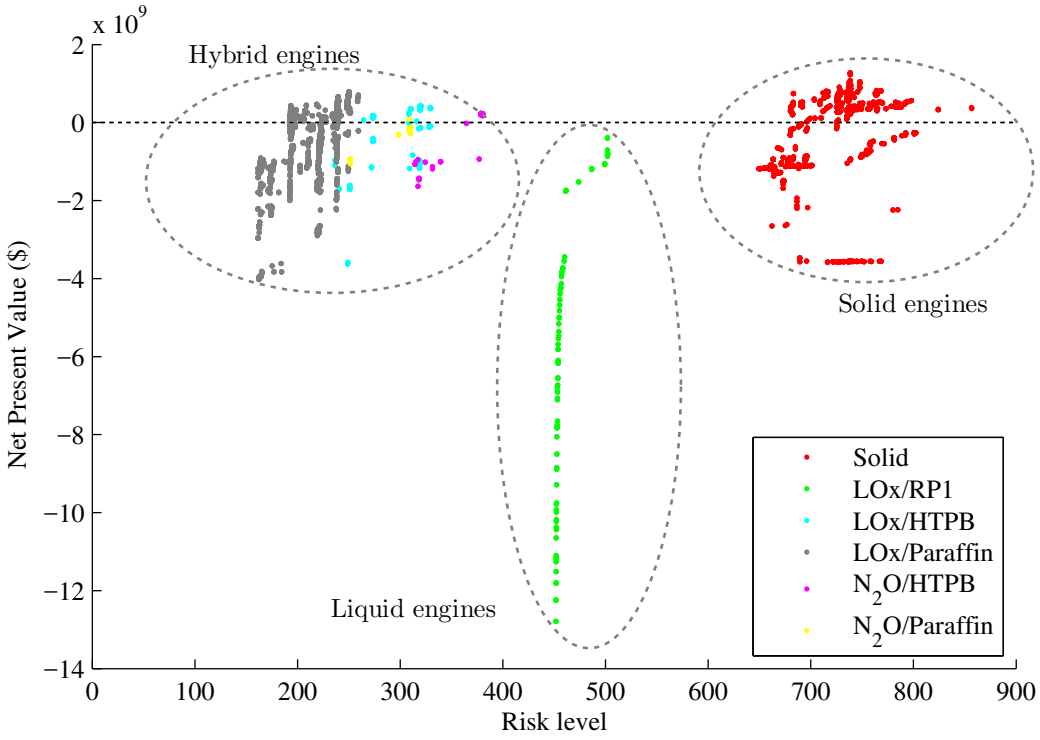



**Figure 128:** Combination of Pareto frontiers colored by propellant

While all mixtures of hybrid propellants are present in the final set, LOx/Paraffin seems to be the best. Indeed, this mixture outscores all other hybrid mixtures in terms of NPV and risk level. Among all the liquid propellant mixtures, only one mixture is present on the final set of vehicles.

Figure 129 shows the non-dominated points colored by type of propellant. Only two mixtures are present on this Pareto frontier: LOx/Paraffin and solid propellant. Solid engines show a higher NPV potential while LOx/Paraffin engines are safer. One can also notice that even though there is a huge gap in risk level between those two mixtures, the

difference in NPV is smaller (only 2% between the best vehicle powered by a LOx/Paraffin engine and the worst powered by a solid engine). Therefore, research should be emphasized on those two mixtures in order to improve future suborbital vehicles.

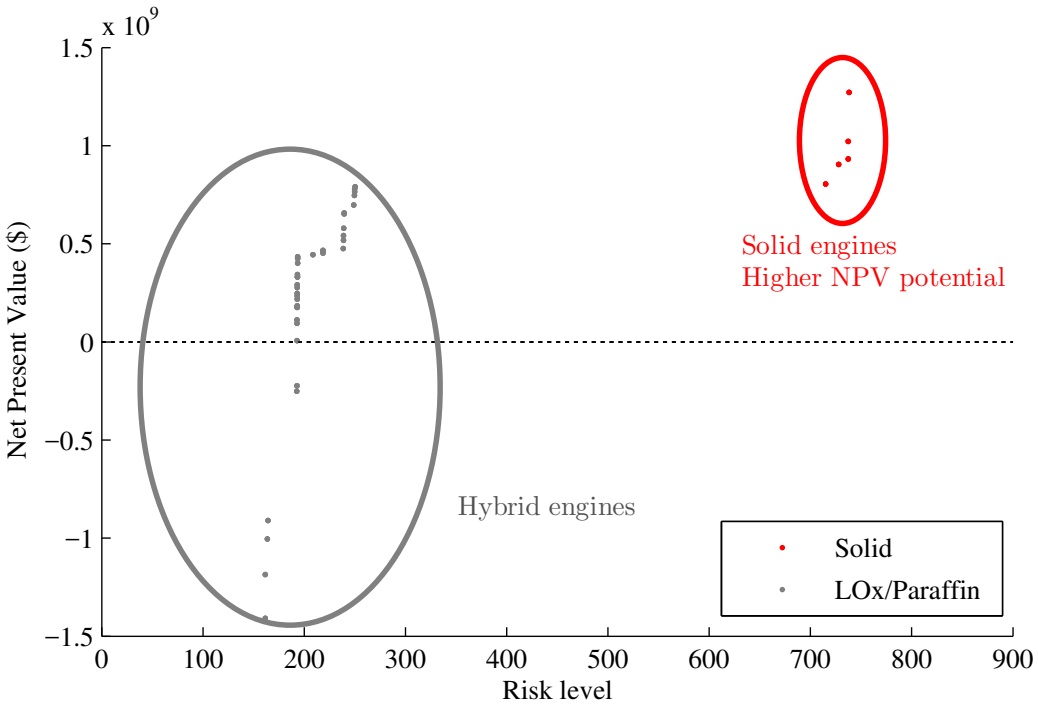



**Figure 129:** Pareto frontier of solutions colored by propellant

Figure 130 provides the combination of all Pareto frontiers from each architecture. One can notice that all architectures are represented in each of the two propellant clusters previously identified. The trends are the same in both clusters: Architecture 1 is the safest but the least profitable, Architecture 4 is the most profitable but the riskiest and Architectures 2 and 3 are in-between. Figure 130 also demonstrates that Architecture 1 is never profitable.

Figure 131 only displays the non-dominated points colored by architecture. It can be noticed that Architecture 2 is not in the final set of optimum solutions. In addition, the trends previously identified among architectures for each cluster remain identical for the final Pareto frontier. Indeed, vehicles built around Architecture 1 appear to be the safest but are not profitable. Concepts built around Architecture 4 are more profitable but riskier.

Figure 132 provides the combination of all Pareto frontiers from each architecture colored

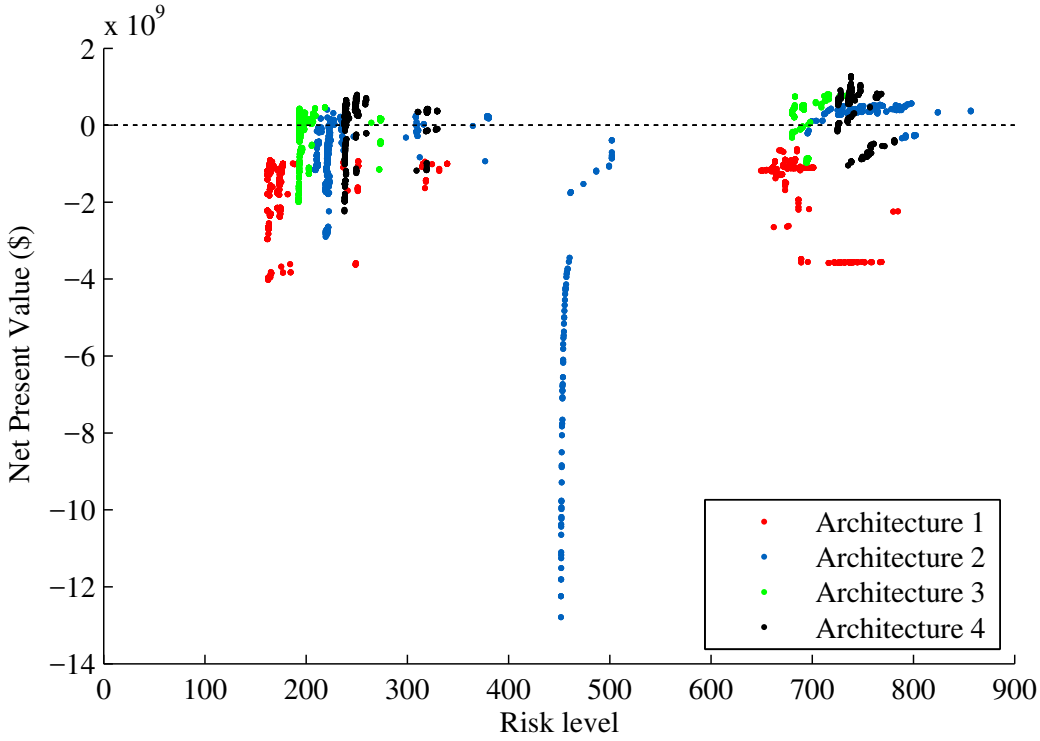



**Figure 130:** Combination of Pareto frontiers colored by architecture

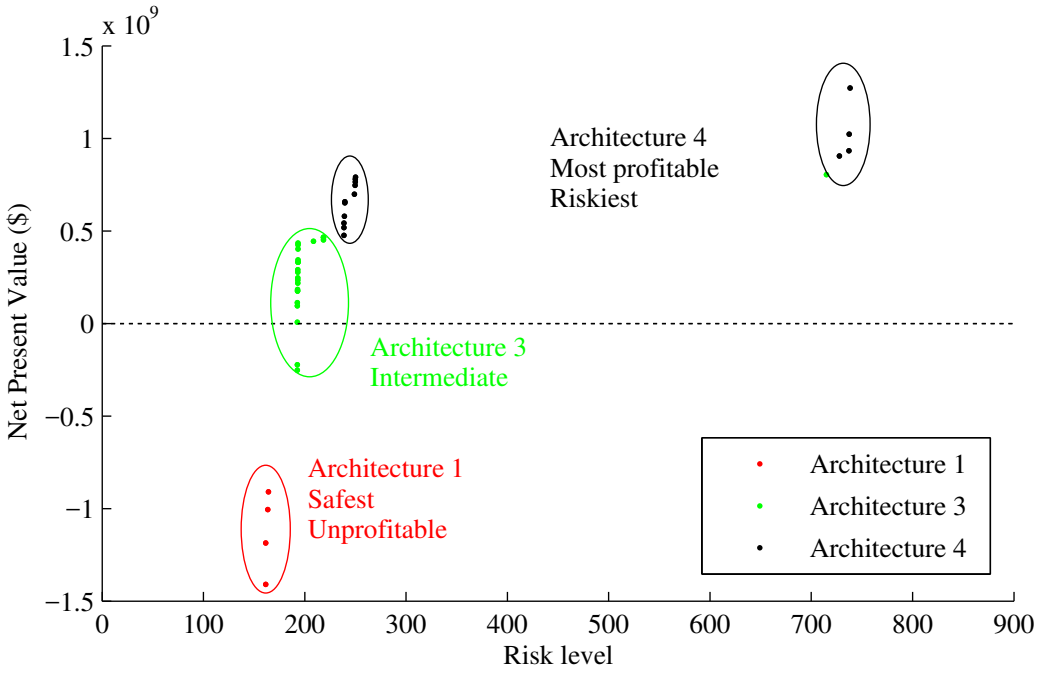



**Figure 131:** Pareto frontier of solutions colored by architecture

by number of passengers. Compared to other parameters, there is a lack of clear trends and all capacities are present in both clusters. Figure 133 only displays the non-dominated

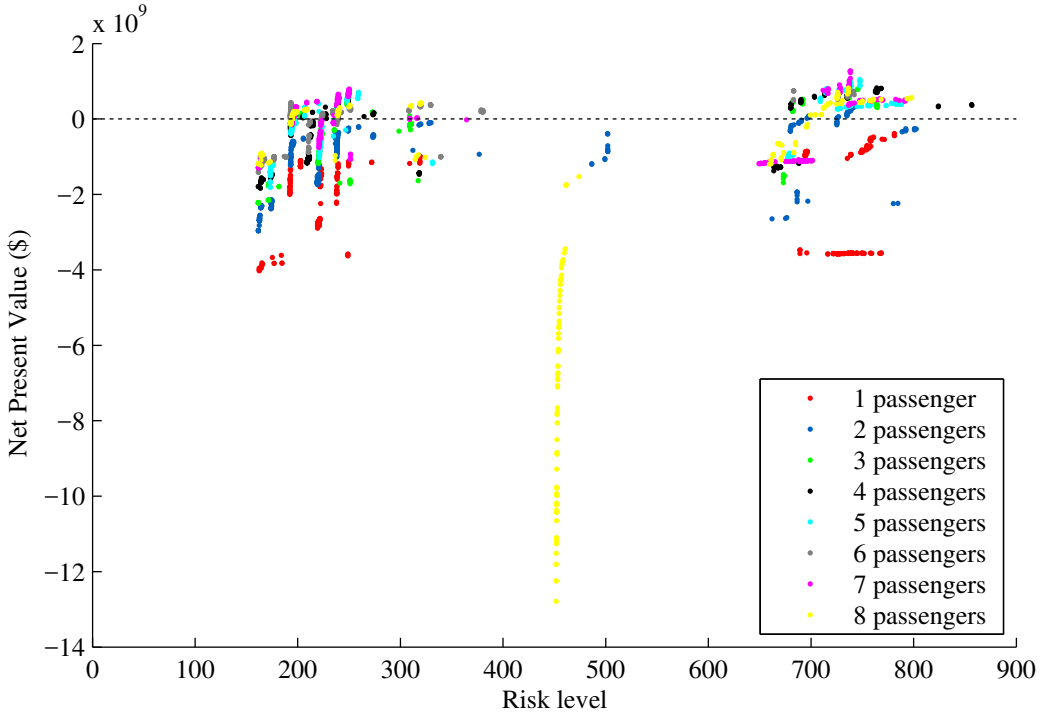



**Figure 132:** Combination of Pareto frontiers colored by capacity

points colored by number of passengers. One can notice that there is an optimum around a six or seven-passenger vehicle and that one and two-passenger vehicles are not present on the final Pareto frontier. As a consequence, increasing the size of the vehicle to improve its economic performance has its limit. The safest vehicles are characterized by an even number of passengers. This can be explained by the presence of the two pilots and the symmetry of the vehicle. Indeed, since vehicles are built around two-seat sections, having an odd number of passengers and pilots would result in sub-optimal solutions.

#### *9.3.4.2 Three-Dimensional Trends*

In order to identify trends in a three-objective environment, ternary plots are used. These barycentric plots display three variables, which sum to a constant value. The advantage of using a ternary plot for depicting dominance is that three variables can be conveniently and continuously plotted in a two-dimensional graph. This allows decision makers to identify trends and promising technological solutions (the ones that occupy a large region of the priority space).

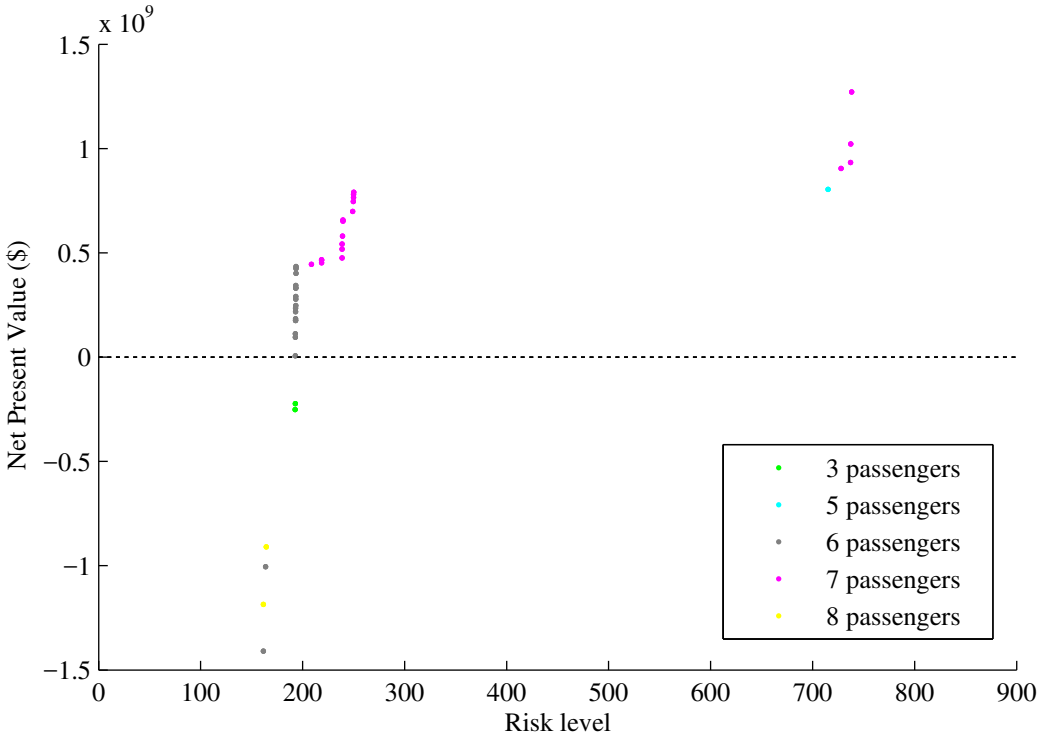



**Figure 133:** Pareto frontier of solutions colored by capacity

Figure 134 displays the regions of the priority space dominated by each architecture. For that purpose, the concept selection methodology developed in Section 7.3 has been used to find the best concept for each combination of design priorities.

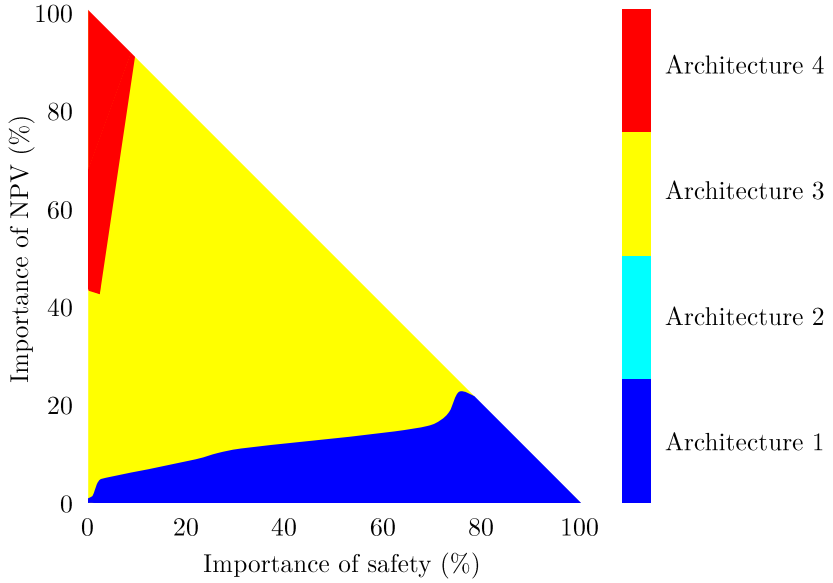



**Figure 134:** Ternary plot colored by architecture

Figure 134 demonstrates that Architecture 3 covers the broader region of the solution space, especially in the interior. In addition, Architecture 1 is suitable in the region where the NPV is not important. Finally, Architecture 4 dominates the region where NPV is the most important.

Figure 135 displays the regions of the priority space for which each propellant is considered to be the most suitable. It clearly shows that LOx/Paraffin (Propellant 6) is the most robust propellant with respect to changes in design priorities. Indeed, it appears to be the best everywhere, except for the region where safety does not have a high importance.

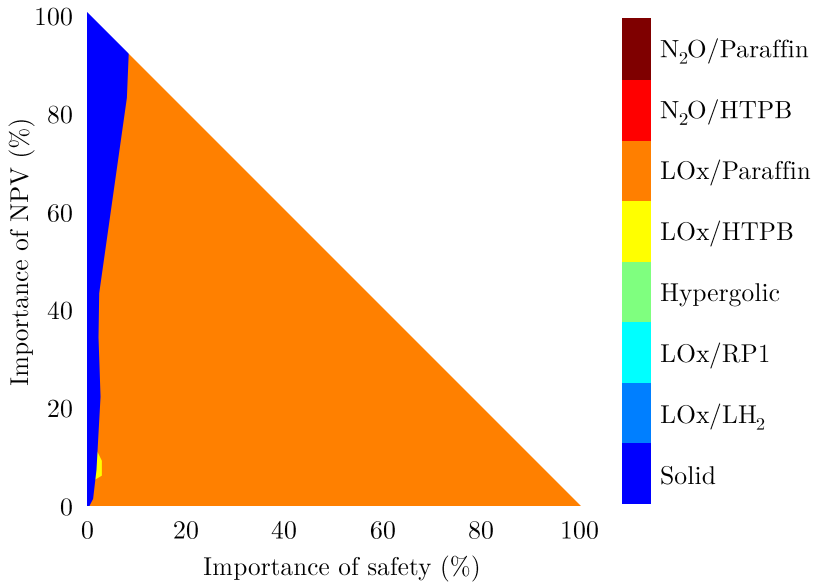



**Figure 135:** Ternary plot colored by propellant

Figures 136 and 137 display the best fleet size and the best ticket price, respectively with respect to different combinations of design priorities. In general, the fleet is characterized by a small number of vehicles (one or two), except for the highly profitable region where a fleet of eight/nine vehicles is needed. For a large fleet, the ticket price can be lower: around \$175k. However, for one and two-vehicle fleets, the optimum ticket price rapidly increases, mostly around \$800k. This is perfectly consistent with the two potential segments identified in Section 9.2.

Figure 138 displays the expected NPV in the priority space. As expected, for higher NPV importance and lower safety and passenger experience importances, the expected NPV

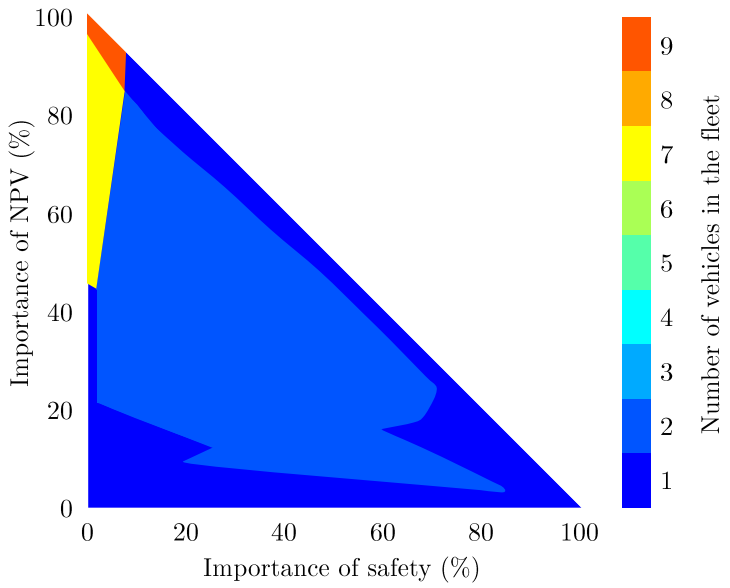



**Figure 136:** Ternary plot colored by fleet size

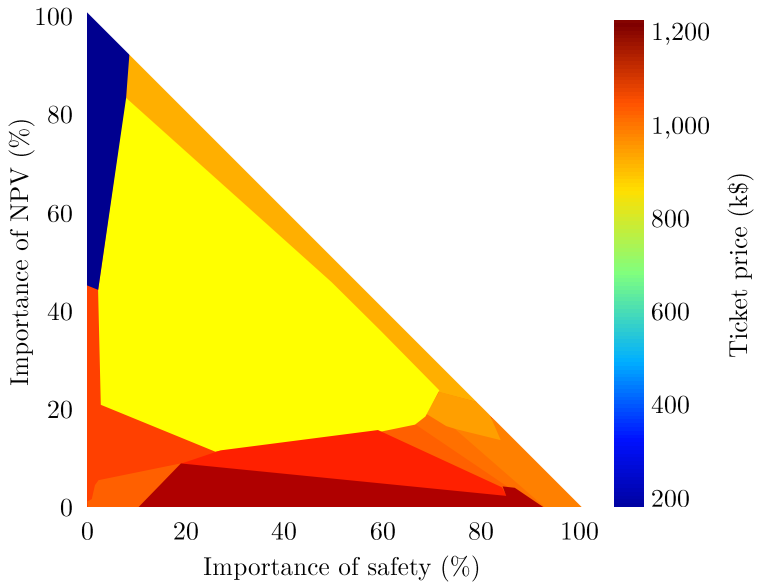



**Figure 137:** Ternary plot colored by ticket price (U.S. k$)

increases. It can also be seen that for most of the combinations, the program can reach a positive NPV, which is promising for the future suborbital market.

While this section aimed at facilitating the identification of the most promising technological solutions as well as some trends about the different concepts, the next section focuses on the identification of dominant families of solutions that can be used as baselines

for further analysis.

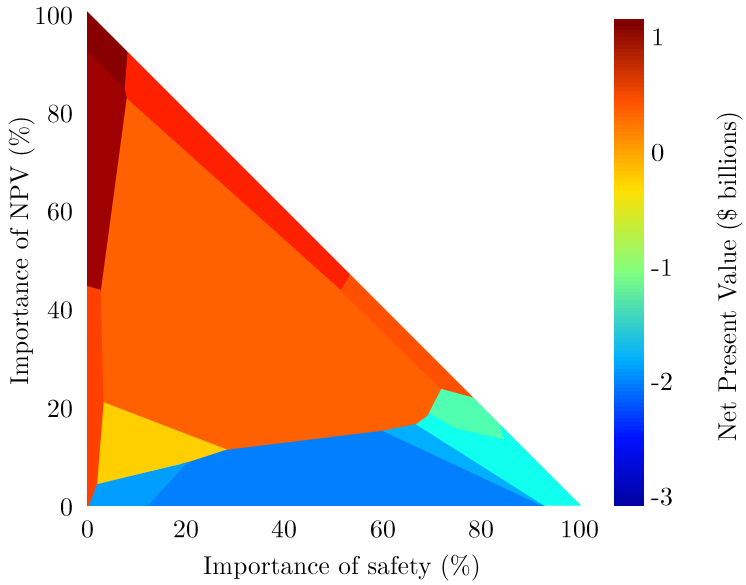



**Figure 138:** Ternary plot colored by NPV

#### *9.3.4.3 Baseline Identification*

In this section, it is assumed that the two main objectives of interest are the expected program NPV and the risk level. As demonstrated in the previous sections, the importance given to each objective highly impacts the optimum vehicle configuration. In order to identify and highlight dominant configurations, a full factorial DoE is performed on the importance of each objective. For each combination, the best concept is selected using the ranking algorithm developed in Section 7.3. Figure 139 shows the evolution of both the program NPV and the safety level with respect to design priorities.

The range covered by the potential values of NPV goes from -$1.3 billion to $1.25 billion. Similarly, the safety level, defined as the maximum risk level minus the risk level of the given concept, varies from 161 to 738.

As shown in Figure 139, there are four dominant concepts defined by four important parameters: architecture, number of passengers, propellant, and number of vehicles in the fleet. The four concepts are described in Table 57.

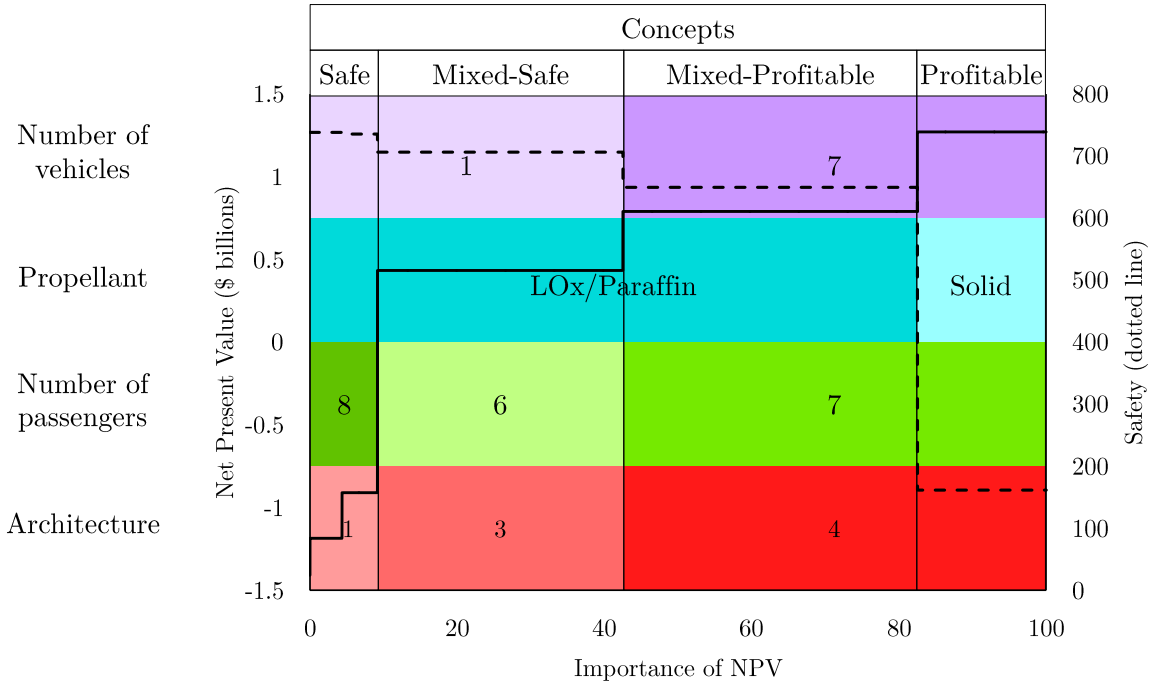



**Figure 139:** Identification of dominant options with respect to design priorities

**Table 57:** Identification of dominant baselines

| **Parameters** | **Safe concept** | **Mixed-safe concept** | **Mixed-profitable concept** | **Profitable concept** |
|---|---|---|---|---|
| **Architecture** | 1 | 3 | 4 | |
| **Number of passengers** | 8 | 6 | 7 | |
| **Propellant** | LOx/Paraffin | | | Solid |
| **Number of vehicles** | 1 | | 7 | |

This table summarizes the key characteristics of the four baselines identified for future suborbital vehicles along with the most promising technological solutions.

As a consequence, this section demonstrates the capabilities of the proposed methodology to perform quantitative trade-offs between objectives, to identify trends among options and technological solutions, and to provide decision makers with the ability to extract dominant configurations. This leads to the validation of the fourth criterion:

**Validation Criterion 4: The methodology provides the capabilities to both perform quantitative trade-offs and identify key trends among all alternatives.**

### 9.3.5 Providing a Complete Picture of the Solutions to Support Informed Decisions

Supporting informed decisions requires to have pertinent and quantitative information about potential solutions readily accessible. To assess the capabilities of the proposed methodology to provide such solutions compared to other methodologies, the number and the type of data output by each approach are compared. The methodologies considered and compared include the typical aircraft design approach, the architecture comparison approach, the architecture optimization approach, and the proposed methodology.

#### *9.3.5.1 Typical Aircraft Design Approach*

The typical aircraft design approach has been applied by numerous authors to design, size, and optimize new suborbital vehicles. In general, they provide well-described solutions along with 3D models or detailed analyses. For instance, Flittie et al. [149] have developed and optimized a ground-launched, single-stage, hybrid propulsion suborbital launch vehicle capable of providing up to ten minutes of microgravity. In this context, a Catia model has been developed based on precise geometric data, as displayed in Figure 140.

Similarly, Gong et al. [177] have designed a Suborbital Reusable Launch Vehicle (SRLV) concept powered by a RBCC engine that takes off and lands horizontally. The design and optimization of the vehicle use a MDO articulated around eight modules: weight/size, parameter study, aerodynamic shape design, RBCC propulsion design, aerodynamics/propulsion integration, trajectory optimization, structural design and sizing, and TPS design and sizing. Each module relies on extremely high-fidelity and accurate analysis codes such as

CFD and FEM techniques. The optimized vehicle is described by a large number of parameters and various models have been generated throughout the design process. However, safety and profitability considerations are often forgotten or made a posteriori. As a consequence, even though the level of detail is sufficient to support well-founded solutions, the type of analyses and data generated do not capture all critical program metrics.

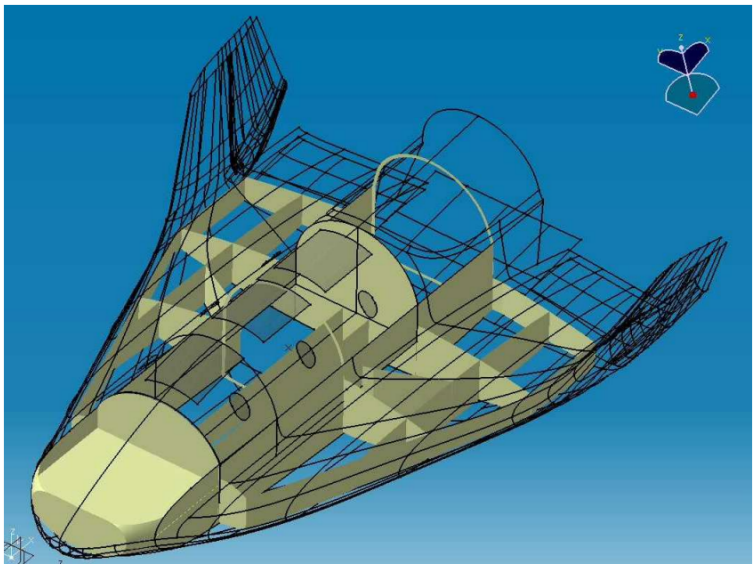


**Figure 140:** Catia model of the HyFlyer [149]

#### *9.3.5.2 Architecture Comparison Approach*

The architecture comparison approach has been used by Sarigul et al. [377] to provide recommendations about the best architecture for manned suborbital reusable launch vehicles. For that purpose, they decomposed the problem into multiple functions: take-off mode, atmospheric entry, landing mode, and propulsion type. For each of these functions, they identified possible solutions based on a review of the state of the art. They also provided qualitative considerations about the performance of each possible solution in order to highlight "four promising architectures that can successfully win the X-Prize:

- Vertical take-off, aerodynamic decelerator
- Vertical take-off, wings with wheel landing
- Some air launch, aerodynamic decelerator

- Some air launch, wings with wheel landing"

They also provide vague considerations about safety and affordability. For example, based on their qualitative analysis, they have concluded that: "there is no clear resolution at this point on the architecture that would have the lowest cost to develop and operate". Finally, no quantitative data was provided about the size, weight, and configuration of the best vehicle(s). Only scaling parameters such as the thrust-to-weight ratio and the propellant mass fraction were considered. As a consequence, this approach does not provide sufficient data to make quantitative, traceable, and informed decisions.

#### *9.3.5.3 Architecture Optimization Approach*

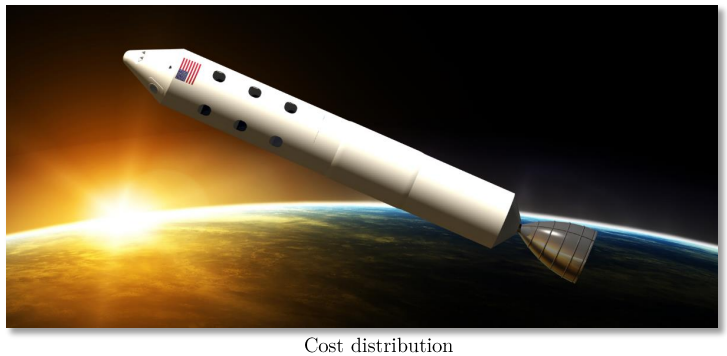
The architecture optimization approach, which consists in optimizing few architectures and manually comparing them, has been developed and used by Villeneuve et al. [455] and applied to the design of launch vehicles. Their analysis focuses on flying performance and life-cycle costs. While this approach represents a first step towards the design of profitable vehicles, it lacks program and financial variables. In addition, the optimization is based on three architectures arbitrarly selected by experts. The number of design variables, and consequently the accuracy of the modeling and simulation environment, does not supply enough information to provide a complete picture of the vehicle from both a physical and a business standpoints. Moreover, their methodology does not provide visualization capabilities, as the concepts are only described using a list of design variables.

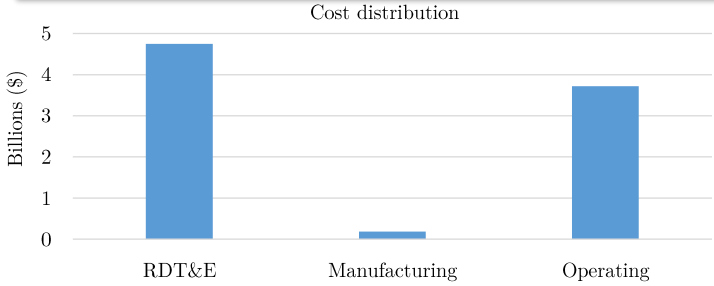
#### *9.3.5.4 Proposed Methodology*

Combining the data obtained from the different analysis modules, the proposed methodology provides detailed information about both the vehicle and the program configurations. In addition, it outputs all the required data needed to make informed decisions in terms of profitability and safety. To demonstrate these capabilities, a complete description output by the proposed methodology is presented in Figures 141 to 144 for the four concepts identified in Section 9.3.4: safe, mixed-safe, mixed-profitable, and profitable concepts.

| | | |
|---|---|---|
| Propulsion | Propellant | LOx/Paraffin |
| | Nozzle expansion ratio | 42 |
| | Chamber pressure | 2 MPa |
| | Rocket engine thrust | 62 kN |
| | Combustion time | 50 s |
| | Jet engines | 0 |
| | Jet engine thrust | - |
| | Bypass ratio | - |
| | Afterburner | - |
| | Turbine Inlet temperature | - |
| Wing | Surface area | - |
| | Sweep angle | - |
| | Aspect ratio | - |
| | Span | - |
| | Taper ratio | - |
| | Thickness-to-chord ratio | - |
| Empennage | Horizontal tail area | - |
| | Horizontal tail sweep angle | - |
| | Horizontal tail aspect ratio | - |
| | Vertical tail area | - |
| | Vertical tail sweep angle | - |
| | Vertical tail aspect ratio | - |
| Fuselage | Diameter | 2.0 m |
| | Length | 19.3 m |
| | Length front part | 1.4 m |
| | Length aft part | 0.6 m |
| | Length propulsion section | 8.71 m |
| | Seat pitch | 1.3 m |

| | | |
|---|---|---|
| Mission | Take-off mode | Vertical |
| | Landing mode | Parachute |
| | Transition altitude | - |
| | Number of passengers | 6 |
| | Number of pilots | 2 |
| | Risk level | 164 |
| | Time in microgravity | 229 s |
| Program | Number of vehicles | 1 |
| | Launch/vehicle/year | 91 |
| | Ticket Price | $ 896,000 |
| | Debt proportion | 10% |
| | Credit Rating | BB+ |
| | IRR | 5.5% |
| | Payback Period | 14.3 years |
| | Program NPV | $ -910 million |
| | Sunk cost | $ 3.9 billion |





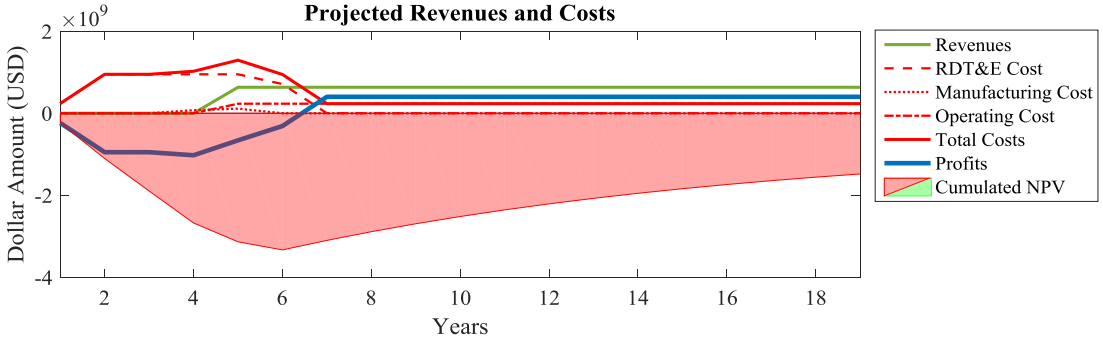
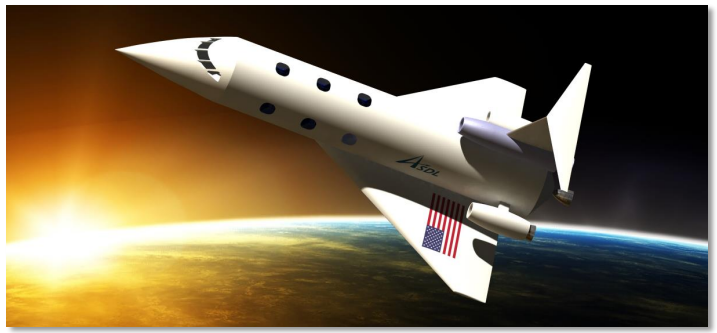
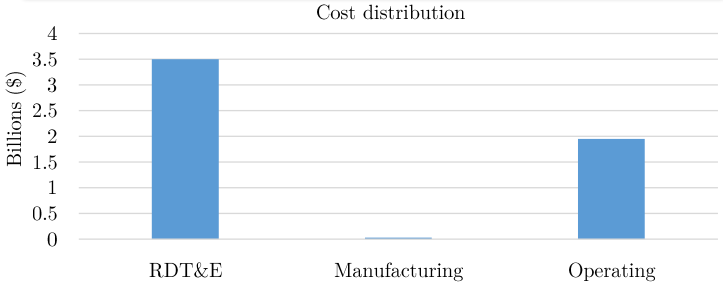




**Figure 141:** Safe concept

| | | |
|---|---|---|
| Propulsion | Propellant | LOx/Paraffin |
| | Nozzle expansion ratio | 90 |
| | Chamber pressure | 2 MPa |
| | Rocket engine thrust | 98 kN |
| | Combustion time | 73 s |
| | Jet engines | 3 |
| | Jet engine thrust | 12 kN |
| | Bypass ratio | 0.46 |
| | Afterburner | No |
| | Turbine Inlet temperature | 2,169 K |
| Wing | Surface area | 33 m$^2$ |
| | Sweep angle | 45.6° |
| | Aspect ratio | 3.3 |
| | Span | 10.43 m |
| | Taper ratio | 0.2 |
| | Thickness-to-chord ratio | 0.10 |
| Empennage | Horizontal tail area | 4.6 m$^2$ |
| | Horizontal tail sweep angle | 16.3° |
| | Horizontal tail aspect ratio | 6.3 |
| | Vertical tail area | 9.2 m$^2$ |
| | Vertical tail sweep angle | 30.1° |
| | Vertical tail aspect ratio | 5.1 |
| Fuselage | Diameter | 2.0 m |
| | Length | 14.7 m |
| | Length front part | 1.8 m |
| | Length aft part | 0.5 m |
| | Length propulsion section | 4.25 m |
| | Seat pitch | 1.3 m |

| | | |
|---|---|---|
| Mission | Take-off mode | Horizontal |
| | Landing mode | Horizontal powered |
| | Transition altitude | 14 km |
| | Number of passengers | 6 |
| | Number of pilots | 2 |
| | Risk level | 193 |
| | Time in microgravity | 241 s |
| Program | Number of vehicles | 1 |
| | Launch/vehicle/year | 104 |
| | Ticket Price | $ 1,015,000 |
| | Debt proportion | 42% |
| | Credit Rating | BB+ |
| | IRR | 11.5% |
| | Payback Period | 10.9 years |
| | Program NPV | $ 434 million |
| | Sunk cost | $ 2.2 billion |





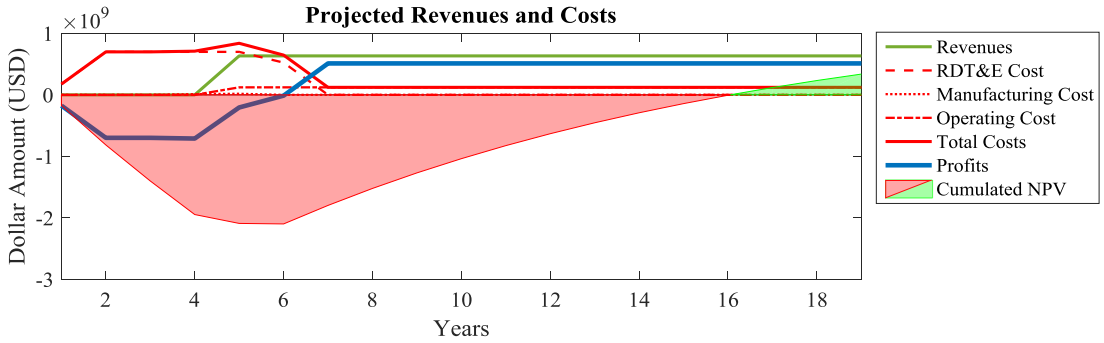
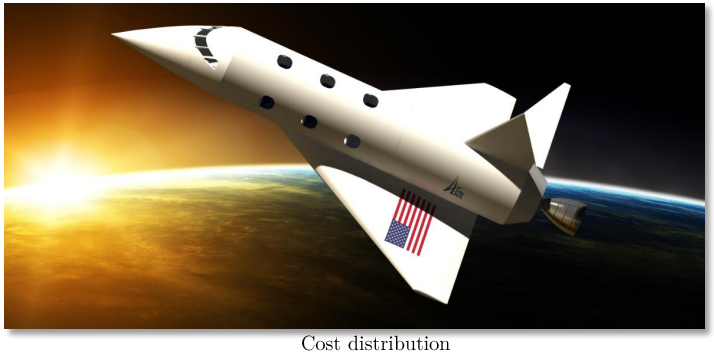
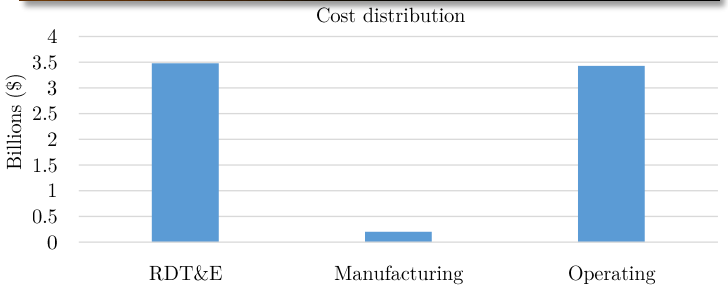




**Figure 142:** Mixed-safe concept

| | | |
|---|---|---|
| Propulsion | Propellant | LOx/Paraffin |
| | Nozzle expansion ratio | 91 |
| | Chamber pressure | 2 MPa |
| | Rocket engine thrust | 101 kN |
| | Combustion time | 73 s |
| | Jet engines | 0 |
| | Jet engine thrust | - |
| | Bypass ratio | - |
| | Afterburner | - |
| | Turbine Inlet temperature | - |
| Wing | Surface area | 31 m$^2$ |
| | Sweep angle | 50.1° |
| | Aspect ratio | 4.1 |
| | Span | 11.22 m |
| | Taper ratio | 0.3 |
| | Thickness-to-chord ratio | 0.13 |
| Empennage | Horizontal tail area | 3.9 m$^2$ |
| | Horizontal tail sweep angle | 26.3° |
| | Horizontal tail aspect ratio | 5.3 |
| | Vertical tail area | 7.8 m$^2$ |
| | Vertical tail sweep angle | 18.7° |
| | Vertical tail aspect ratio | 6.0 |
| Fuselage | Diameter | 2.0 m |
| | Length | 13.7 m |
| | Length front part | 1.2 m |
| | Length aft part | 0.7 m |
| | Length propulsion section | 4.35 m |
| | Seat pitch | 1.3 m |

| | | |
|---|---|---|
| Mission | Take-off mode | Air launched |
| | Landing mode | Gliding |
| | Transition altitude | 13 km |
| | Number of passengers | 7 |
| | Number of pilots | 1 |
| | Risk level | 250 |
| | Time in microgravity | 237 s |
| Program | Number of vehicles | 7 |
| | Launch/vehicle/year | 104 |
| | Ticket Price | $ 170,226 |
| | Debt proportion | 31% |
| | Credit Rating | BB+ |
| | IRR | 13.2% |
| | Payback Period | 10.1 years |
| | Program NPV | $ 791 million |
| | Sunk cost | $ 2.7 billion |





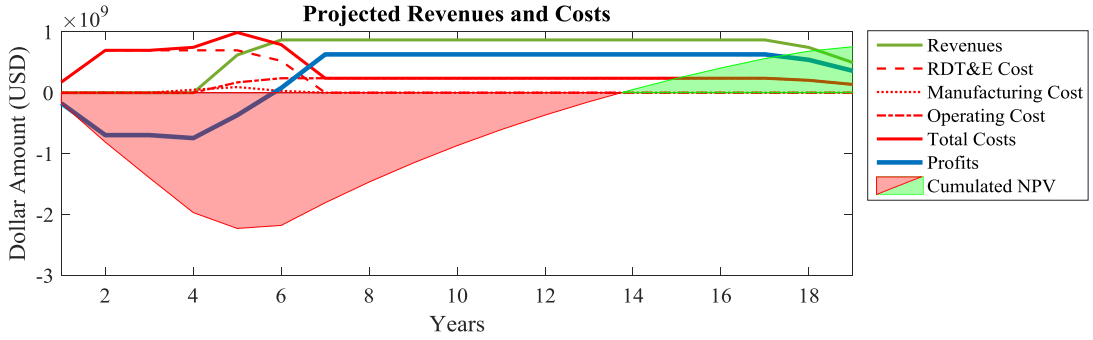
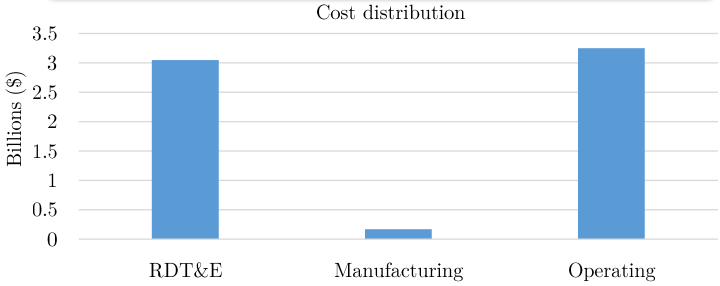




**Figure 143:** Mixed-profitable concept

| | | |
|---|---|---|
| Propulsion | Propellant | Solid |
| | Nozzle expansion ratio | 91 |
| | Chamber pressure | 2 MPa |
| | Rocket engine thrust | 91 kN |
| | Combustion time | 154 s |
| | Jet engines | 0 |
| | Jet engine thrust | - |
| | Bypass ratio | - |
| | Afterburner | - |
| | Turbine Inlet temperature | - |
| Wing | Surface area | 29.7 m$^2$ |
| | Sweep angle | 41.2° |
| | Aspect ratio | 2.2 |
| | Span | 8.81 m |
| | Taper ratio | 0.3 |
| | Thickness-to-chord ratio | 0.13 |
| Empennage | Horizontal tail area | 2.3 m$^2$ |
| | Horizontal tail sweep angle | 17.1° |
| | Horizontal tail aspect ratio | 5.2 |
| | Vertical tail area | 4.7 m$^2$ |
| | Vertical tail sweep angle | 33.4° |
| | Vertical tail aspect ratio | 3.2 |
| Fuselage | Diameter | 2.0 m |
| | Length | 13.5 m |
| | Length front part | 1.0 m |
| | Length aft part | 0.9 m |
| | Length propulsion section | 5.71 m |
| | Seat pitch | 1.3 m |

| | | |
|---|---|---|
| Mission | Take-off mode | Air launched |
| | Landing mode | Gliding |
| | Transition altitude | 14 km |
| | Number of passengers | 7 |
| | Number of pilots | 1 |
| | Risk level | 738 |
| | Time in microgravity | 195 s |
| Program | Number of vehicles | 7 |
| | Launch/vehicle/year | 104 |
| | Ticket Price | $ 170,000 |
| | Debt proportion | 40% |
| | Credit Rating | BB+ |
| | IRR | 16.1% |
| | Payback Period | 9.3 years |
| | Program NPV | $ 1,270 million |
| | Sunk cost | $ 2.3 billion |



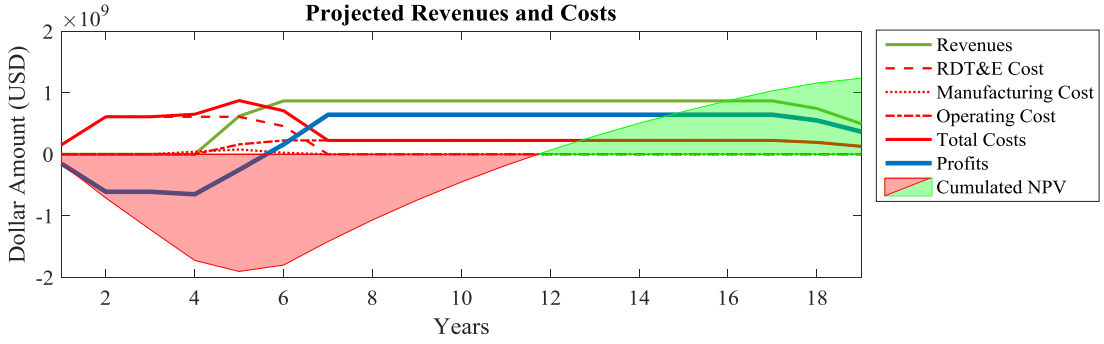




**Figure 144:** Profitable concept

The description provides all necessary decision metrics in terms of vehicle, economic, and safety performance. In particular, the vehicle performance is described by the time spent in microgravity, the seat pitch, etc. The economic performance is modeled by key financial metrics used by investors to make important decisions: the program Internal Rate Of Return (IRR), the expected NPV, the payback period, the sunk cost, the ticket price, etc. Finally, the safety level of the vehicle is defined by a specifically developed scale that evaluates the risk of catastrophic failure. As a consequence, the proposed methodology provides decision makers with all the important metrics necessary to make informed and traceable decisions.

In addition, as described in Section 7.4, a dynamic and parametric visualization platform is developed to assist designers in their analyses, selections, and decisions. Indeed, the 3D model provides designers with a high-level overview of the selected vehicle(s). By linking the optimization results generated by the Matlab code to the CAD software, the Excel dashboard helps multi-disciplinary teams work together and reach a consensus about the decisions. As presented below, the 3D Catia model can be used for various and specific purposes:

- Advertising and marketing images: the 3D view can be used for advertising and marketing purposes as it presents the global aspect of the vehicle.

**Figure 145:** Example of 3D marketing image that can be generated

Indeed, this platform provides decision makers with communication contents that can be used to make promotional images or movies for fundraising or marketing purposes. In addition, exploded views enable an exploration of all the subsystems and are also often used for marketing purposes. In particular, it helps companies highlight the key success factors and competitive advantages of their products. An example of such images is displayed in Figure 145.

- Further analysis: the developed Catia model can also be used to perform further analyses on the vehicle, as displayed in Figure 146.

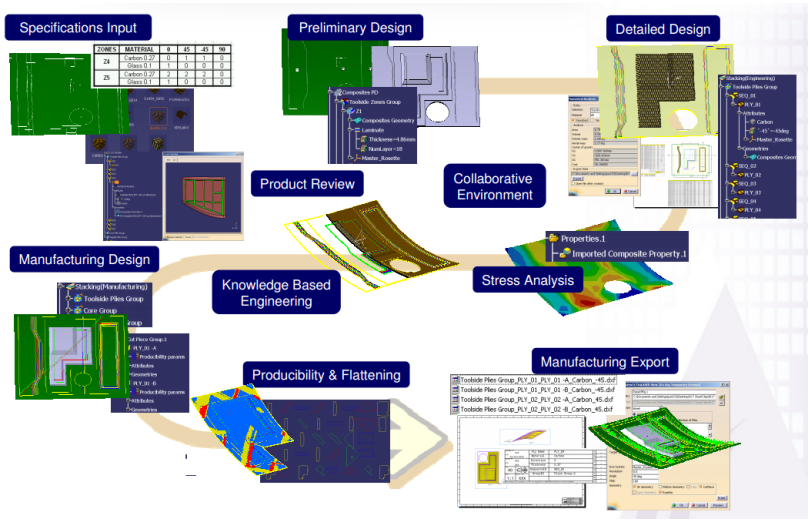



**Figure 146:** Possible analysis using the Catia model [128]

Indeed, based on the specifications provided by the Matlab optimization, preliminary and detailed designs can be conducted using Catia. A structural and stress analysis module enables mechanical analyses on the different subsystems and on the system as a whole. Hence, designers benefit from a preview of the structural resistance at a conceptual level of the design. In addition, 3-views can be created to provide detailed drawings. Section cuts also help visualize the interior subsystems and of the layout of the different parts. Finally, producibility and manufacturability considerations can also be made. An example of detailed drawings available is presented in Figure 147.

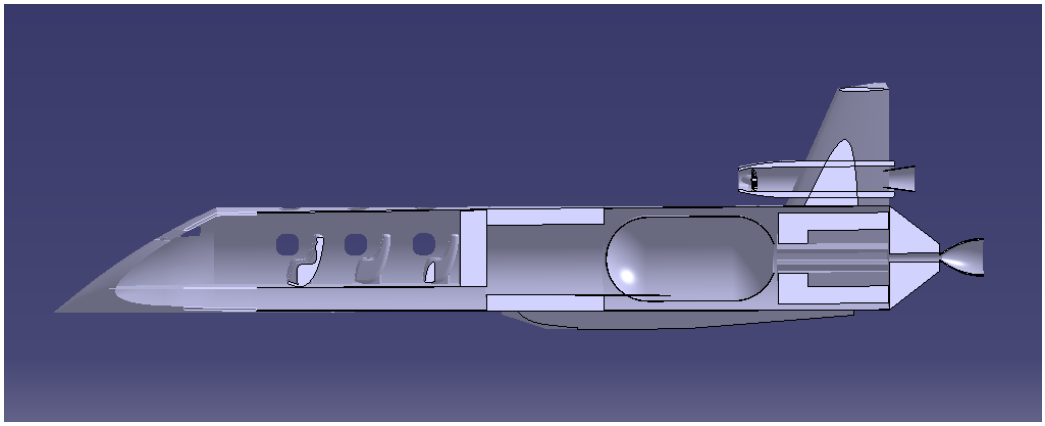


**Figure 147:** Vertical lateral section of the vehicle

- Rapid prototyping: the developed parametric CAD model can be used for rapid prototyping. Indeed, the Catia model generated using the visualization dashboard can be easily converted into a stereolithography (stl) file, which can be read by most 3D printers. Examples of printed concepts are displayed in Figures 148 and 149. They have been generated with the 3D-printer available at the ASDL.

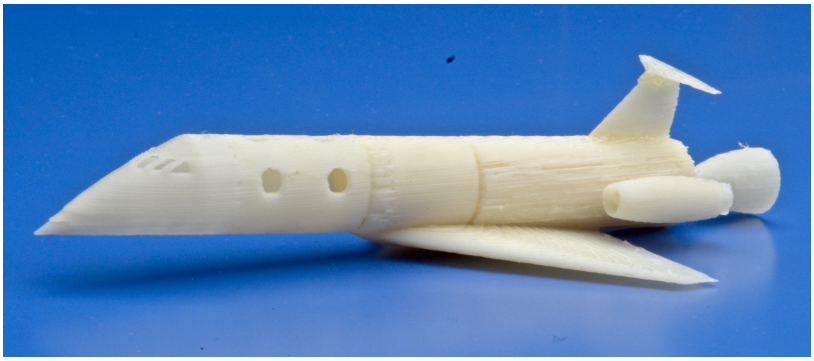


**Figure 148:** 3D model of a winged-body vehicle with jet engines

All these observations result in the validation of the fifth criterion:

**Validation Criterion 5: The proposed methodology provides decision makers with a complete picture from both a physical and a business standpoints, hence supporting informed decision-making.**

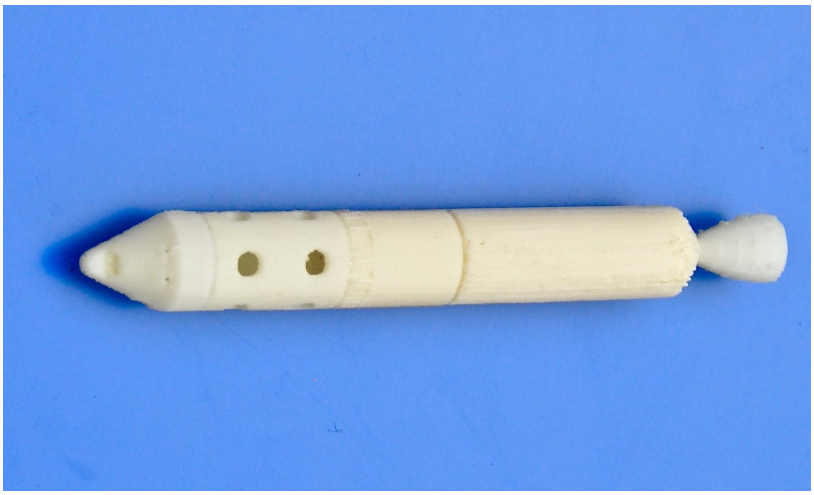


**Figure 149:** 3D model of a slender-body vehicle

### 9.3.6 Performing Trade-Off Analyses Between Performance and Robustness

When dealing with emerging markets, the risk taken by stakeholders and decision makers is usually important as the requirements' uncertainty is high. To alleviate this risk, there is a need for concepts to be robust to changes in requirements. However, increasing the robustness tends to decrease the overall performance of the vehicle. As a consequence, decision makers must have the capabilities to identify the key uncertainty sources, trade performance against robustness, select a robust concept, and assess the risk related to each decision.

#### *9.3.6.1 Identifying the Key Uncertainty Sources*

When starting a program, it is important to identify the main sources of risk. For that purpose, a Latin Hypercube DoE is performed on the five uncertainty sources described in Section 4: maximum altitude, maximum acceptable load factor, development time, demand, and risk of catastrophic failure. Then, based on this DoE, a sensitivity analysis is performed to highlight the big hitters. Figure 150 presents the sensitivity of the standard deviation of the NPV with respect to the uncertainty sources. In particular, the relative impact of the three key sources is presented. One can notice that uncertainty in mission requirements has a higher impact than the one linked to the program.

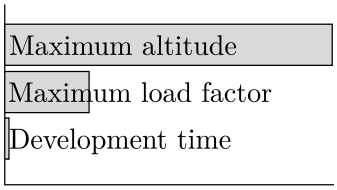



**Figure 150:** Sensitivity analysis of the uncertainty sources

#### *9.3.6.2 Providing an Overview of the Time-Dependent Robustness of the Vehicles*

Figure 151 displays the trade-offs that must be made between the concepts' performance, represented by the NPV, and its robustness (modeled by its standard deviation) at different points in time. Indeed, as discussed in Section 9.2.2, uncertainty in requirements decreases with time. As a consequence, the degree of requirements' uncertainty in Year 1 is higher than the one in Year 10. This results in an overall decrease of performance uncertainty over time. Figure 151 provides the non-dominated concepts for multiple years into the program. These time-dependent Pareto frontiers clearly show that concepts with a higher NPV are riskier as the standard deviation of their NPV is higher.

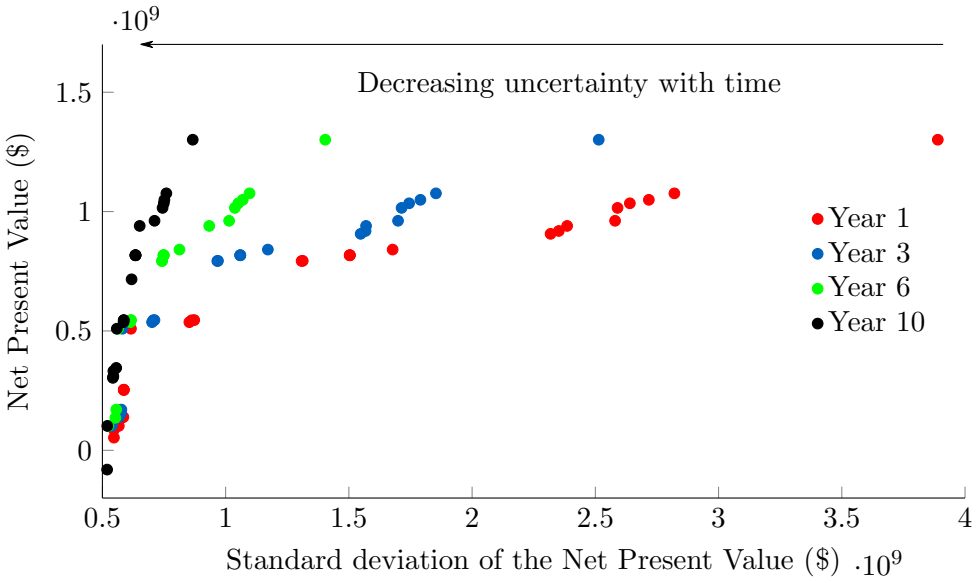



**Figure 151:** Uncertainty evolution with time

As expected, the standard deviation of the solutions tends to decrease with time. Indeed, for a fixed value of NPV equal to $1 billion, its corresponding standard deviation decreases from $2.7 billion in Year 1 to $1.7 billion in Year 3, and to $1 billion in Year 6. Over

time, the new Pareto frontiers tend to dominate the previous ones and offer a better set of solutions with respect to the two objectives: NPV and standard deviation of the NPV.

In addition, the proposed methodology also allows decision makers to identify points of interest in the Pareto frontier at different points in time. In particular, in early stages of the market, the Pareto frontier is characterized by a flat region around a NPV of $800 million (Figure 151). This shows that, in this region, a small improvement in NPV results in an important deterioration of its standard deviation. Hence, designers might want to select a design in the left-hand side of this region. Similarly, all Pareto frontiers are characterized by a sharp increase in NPV for small standard deviation concepts. This shows that a great improvement in NPV can be made in this region for a small deterioration in standard deviation (robustness). Moreover, it can be noted that for these concepts, time does not help decrease uncertainty. Indeed, in order to decrease the program risk, the number of vehicles built and the number of flights are to be reduced. Hence, even if there are important changes in demand or in requirements, the impact on the NPV will not be too important.

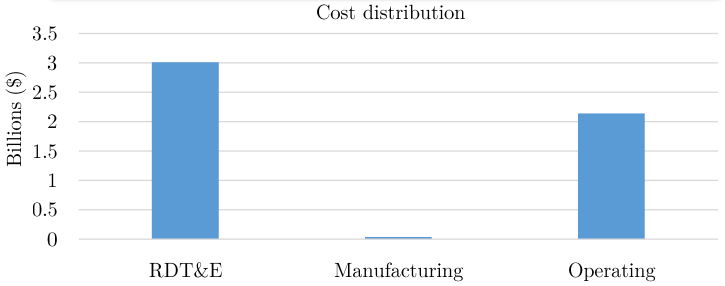
This analysis is crucial for decision makers as it helps them trade the overall program risk against the expected performance.

#### *9.3.6.3 Selecting a Robust Concept*

The application of the selection algorithm developed in Section 7.3 allows decision makers to easily include robustness in the set of decision criteria. In particular, the same importance is given to expected NPV and robustness. The best concept is built around Architecture 3 with one jet engine and carries five passengers with one pilot. The detailed description of the robust concept is provided in Figure 152.

| | | |
|---|---|---|
| Propulsion | Propellant | Solid |
| | Nozzle expansion ratio | 60 |
| | Chamber pressure | 3 MPa |
| | Rocket engine thrust | 53 kN |
| | Combustion time | 76 s |
| | Jet engines | 1 |
| | Jet engine thrust | 16 kN |
| | Bypass ratio | 0.54 |
| | Afterburner | Yes |
| | Turbine Inlet temperature | 2,012 K |
| Wing | Surface area | 29 m$^2$ |
| | Sweep angle | 43.5° |
| | Aspect ratio | 3.3 |
| | Span | 9.8 m |
| | Taper ratio | 0.3 |
| | Thickness-to-chord ratio | 0.11 |
| Empennage | Horizontal tail area | 4.1 m$^2$ |
| | Horizontal tail sweep angle | 26.6° |
| | Horizontal tail aspect ratio | 6.1 |
| | Vertical tail area | 8.2 m$^2$ |
| | Vertical tail sweep angle | 27.4° |
| | Vertical tail aspect ratio | 5.6 |
| Fuselage | Diameter | 2.0 m |
| | Length | 12.7 m |
| | Length front part | 1.3 m |
| | Length aft part | 0.5 m |
| | Length propulsion section | 4.06 m |
| | Seat pitch | 1.3 m |

| | | |
|---|---|---|
| Mission | Take-off mode | Horizontal |
| | Landing mode | Horizontal powered |
| | Transition altitude | 12 km |
| | Number of passengers | 5 |
| | Number of pilots | 1 |
| | Risk level | 733 |
| | Time in microgravity | 229 s |
| Program | Number of vehicles | 2 |
| | Launch/vehicle/year | 81 |
| | Ticket Price | $ 814,000 |
| | Debt proportion | 35% |
| | Credit Rating | BB+ |
| | IRR | 13.8% |
| | Payback Period | 9.9 years |
| | Program NPV | $ 764 million |
| | Sunk cost | $ 2.1 billion |



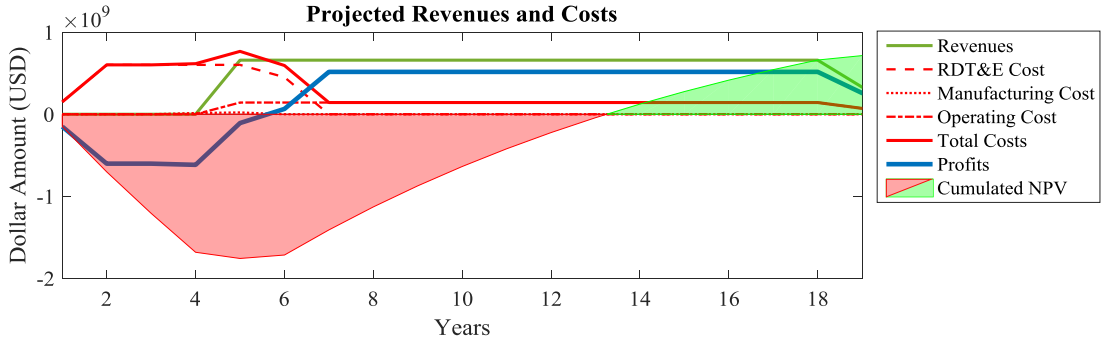




**Figure 152:** Robust concept

#### 9.3.6.4 *Comparing the Performance and Evaluating the Robustness of Selected Concepts*

Once several concepts of interest have been selected for further analysis, the methodology enables trades to be conducted between performance and standard deviation of the performance. In particular, Figure 153 displays both the evolution of the probability to have a positive NPV and the standard deviation of the NPV with time, for the five selected concepts: safe, mixed-safe, mixed-profitable, profitable, and robust.

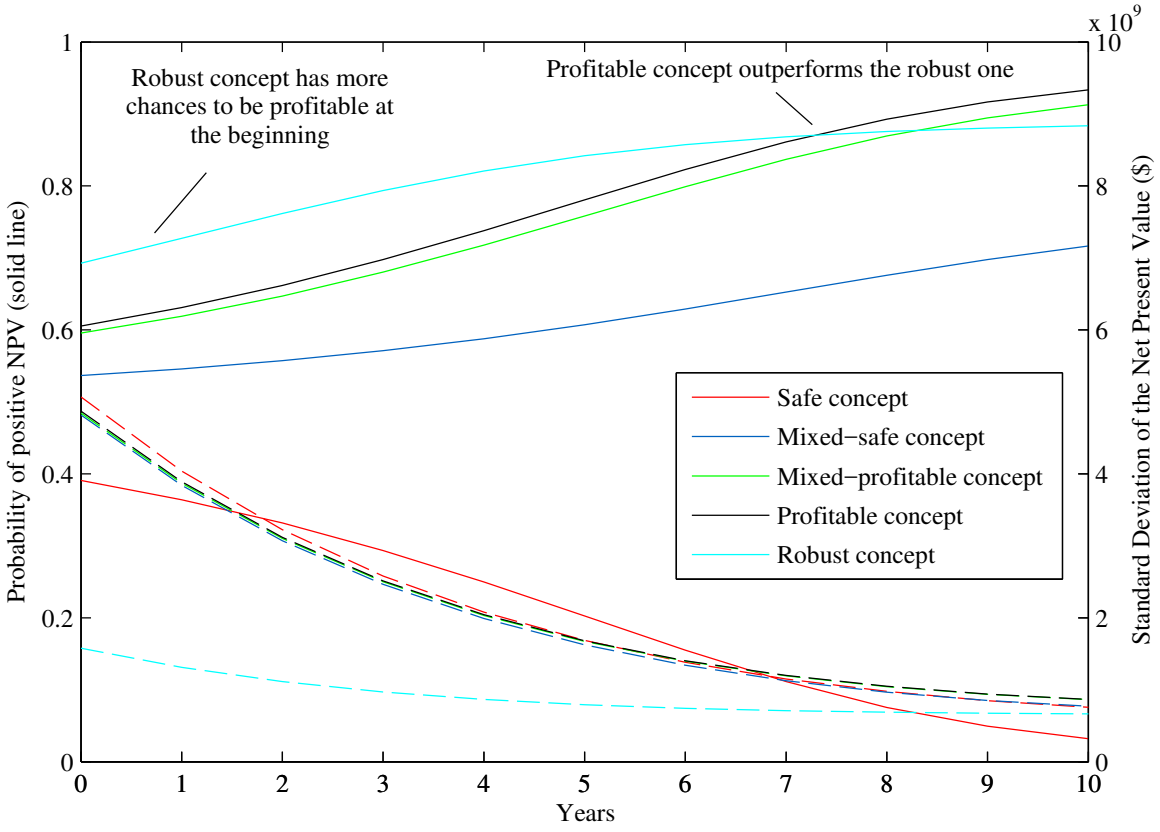



**Figure 153:** Probability of positive NPV and standard deviation over time

Figure 153 shows that the standard deviation decreases over time for all the concepts considered. In addition, one can notice that, while the safe concept has the highest standard deviation at the beginning, its standard deviation becomes smaller than the ones of other concepts when the requirements start to freeze. At that time, the profitable concept becomes the most uncertain.

Figure 153 also highlights the fact that the probability of having a positive NPV does not

always increase over time. Indeed, for concepts with an initial low probability of having a positive NPV (safe concept), this probability decreases over time. On the contrary, concepts with a high probability of having a positive NPV at the beginning tend to see this probability increase over time. This can be explained by the fact that concepts with an initial low probability of making profit have a negative average NPV. As uncertainty decreases, the standard deviation tends to be smaller and the distribution tends to draw closer around this negative value, consequently decreasing the probability of having a positive NPV. The opposite phenomenon occurs for concepts with a high probability of having a positive NPV.

Finally, while the profitable concept has the highest probability of having a positive NPV at the end (when requirements' uncertainty is low), it only outperforms the robust concept after Year 7. This analysis provides the best trade-off between risk and return in terms of NPV at each point in time. This information is crucial when selecting the best concept in the presence of a large amount of uncertainty in requirements. Indeed, using the proposed methodology allows decision makers to highly increase the probability of designing a profitable concept under uncertainty. In particular, at Year 1, the probability of designing a profitable suborbital program is increased by around 10% when robustness is included in the set of decision criteria.

#### *9.3.6.5 Assessing the Impact of Including Robustness as a Decision Criterion*

For simplicity, the previous concept selection only relied on two objectives: NPV and standard deviation of the NPV. However, in complex solution spaces, more than one objective have to be traded against robustness. In this section, the methodology is used to select the best concept with respect to multiple objectives. Table 58 summarizes the impact of including more objectives on the overall characteristics of the vehicles.

Based on these data, the radar plot presented in Figure 154 has been created. The data have been normalized from 0 (worst value of a given objective) to 1 (best value of a given objective) in order to compare each objective.

**Table 58:** Impact of including more decision criteria on the selected concepts

| Parameters | Safe | Profitable | Profitable-safe | Mixed |
|---|---|---|---|---|
| **Importance NPV (%)** | 0 | 100 | 50 | 33 |
| **Importance safety (%)** | 100 | 0 | 50 | 33 |
| **Importance standard deviation (%)** | 0 | 0 | 0 | 33 |
| **NPV ($ billion)** | -1.4 | 1.3 | 0.4 | 0.3 |
| **Risk level** | 162 | 738 | 193 | 193 |
| **Standard deviation of NPV ($ billion)** | 4.04 | 3.89 | 3.84 | 1.06 |

Figure 154 shows the importance of including robustness in order to decrease the standard deviation of the selected concept. Figure 154 also demonstrates that including robustness highly decreases the standard deviation of the best concept without deteriorating the expected NPV and the safety by more than several percents. Hence, using the proposed methodology allows decision makers to highly reduce the overall program risk without deteriorating its performance. Figure 155 presents the description of the mixed concept.

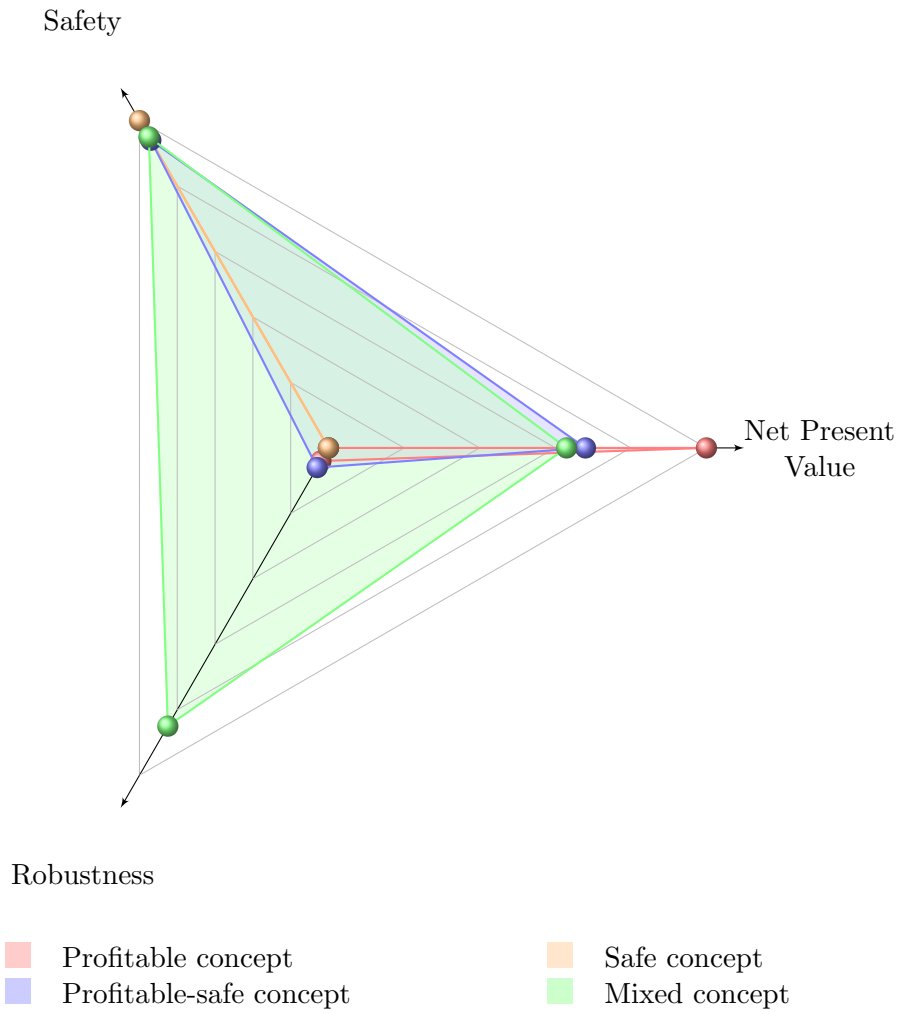
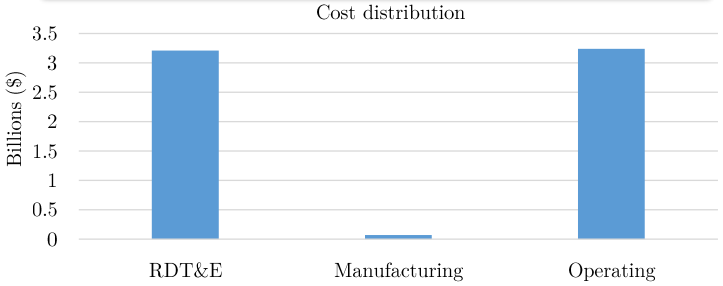



**Figure 154:** Trade-off between performance metrics and robustness

| | | |
|---|---|---|
| Propulsion | Propellant | LOx/Paraffin |
| | Nozzle expansion ratio | 89 |
| | Chamber pressure | 2 MPa |
| | Rocket engine thrust | 71 kN |
| | Combustion time | 81 s |
| | Jet engines | 3 |
| | Jet engine thrust | 19 kN |
| | Bypass ratio | 0.50 |
| | Afterburner | No |
| | Turbine Inlet temperature | 1,907 K |
| Wing | Surface area | 28 m$^2$ |
| | Sweep angle | 47.3° |
| | Aspect ratio | 3.4 |
| | Span | 9.72 m |
| | Taper ratio | 0.2 |
| | Thickness-to-chord ratio | 0.10 |
| Empennage | Horizontal tail area | 5.2 m$^2$ |
| | Horizontal tail sweep angle | 27.5° |
| | Horizontal tail aspect ratio | 5.9 |
| | Vertical tail area | 10.3 m$^2$ |
| | Vertical tail sweep angle | 21.7° |
| | Vertical tail aspect ratio | 7.1 |
| Fuselage | Diameter | 2.0 m |
| | Length | 11.3 m |
| | Length front part | 2.4 m |
| | Length aft part | 0.7 m |
| | Length propulsion section | 4.45 m |
| | Seat pitch | 1.3 m |

| | | |
|---|---|---|
| Mission | Take-off mode | Horizontal |
| | Landing mode | Horizontal powered |
| | Transition altitude | 15 km |
| | Number of passengers | 3 |
| | Number of pilots | 1 |
| | Risk level | 203 |
| | Time in microgravity | 228 s |
| Program | Number of vehicles | 3 |
| | Launch/vehicle/year | 87 |
| | Ticket Price | $ 839,972 |
| | Debt proportion | 21% |
| | Credit Rating | BB+ |
| | IRR | 9.9% |
| | Payback Period | 11.4 years |
| | Program NPV | $ 94 million |
| | Sunk cost | $ 2.4 billion |



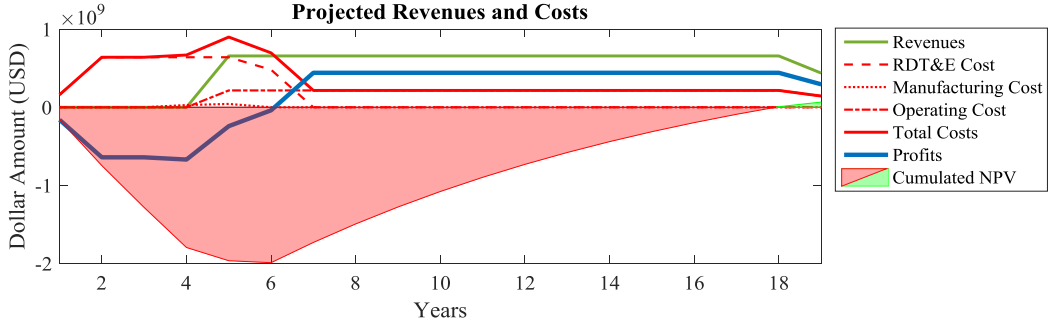




**Figure 155:** Mixed concept

This section illustrated the ability of the proposed methodology to perform important trade-offs between robustness and performance. In particular, its benefits in alleviating a program risk due to customer uncertainty have been clearly demonstrated. This leads to the validation of the sixth criterion:

**Validation Criterion 6: The methodology provides the ability to easily trade performance against robustness in order to alleviate the program risk.**

### 9.3.7 Supporting Go/No-Go Decisions

One of the key capabilities of the developed methodology is to support informed go/no-go decisions in the presence of evolving uncertainty in requirements. This section discusses and illustrates such capabilities by evaluating the best possible performance under a maximum risk, estimating the best start date of the program, and conducting time-dependent risk and return analyses.

#### *9.3.7.1 Evaluating the Best Possible Performance under a Maximum Risk Limit*

Usually, decision makers tend to agree on a maximum program risk and then select the best corresponding concept. In this section, the capabilities of the proposed methodology in supporting such tasks is evaluated. For that purpose, a maximum standard deviation of $1 billion is arbitrarily fixed and the best concept is selected over time that meets this constraint. This aims at helping decision makers in evaluating the potential ROI of a specific program while accepting a given risk.

For that purpose, Figure 156 provides the evolution of both the maximum potential program NPV and its corresponding standard deviation over time. Indeed, at each point in time (under a given degree of uncertainty), a constrained optimization problem is solved to find the concept with the highest expected NPV whose standard deviation remains within the limits.

As expected, the maximum potential NPV increases over time. The curve follows a step function, where each jump corresponds to a change in the optimum concept. Indeed, as time increases, more profitable concepts become feasible with respect to the uncertainty constraint. Similarly, the standard deviation curve follows a saw-tooth pattern. For a given

concept (flat region on the maximum NPV curve), the corresponding standard deviation decreases with time. When a better concept is found, there is a jump in both curves and the standard deviation curve reaches its maximum allowed value before decreasing again. The large number of jumps in the curves shows that, when the degree of uncertainty decreases, the optimum concept rapidly changes. As a consequence, it is crucial to benefit from a flexible methodology that supports decisions over time.

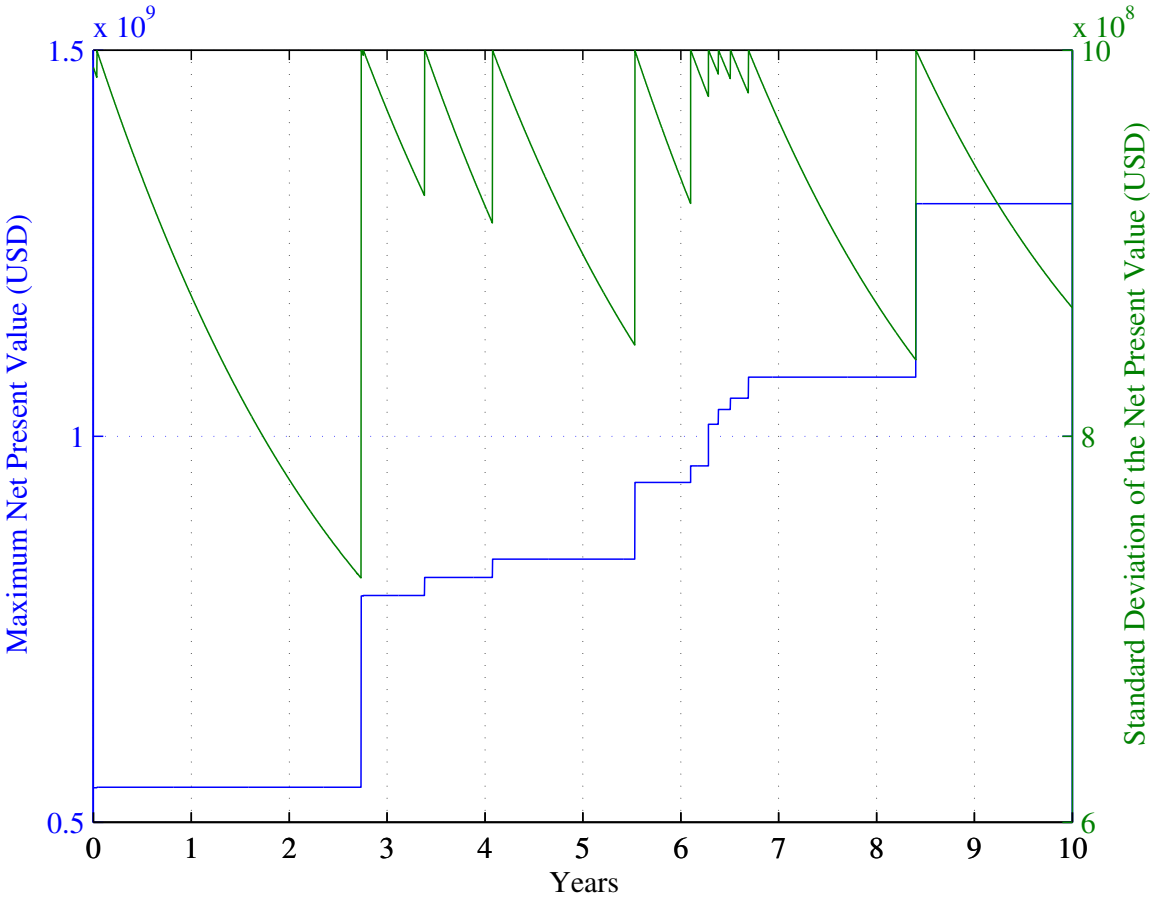



**Figure 156:** Optimum NPV under a maximum standard deviation of $1 billion

Figure 157 combines both the standard deviation and the expected return to evaluate the corresponding probability of having a positive NPV.

As expected, on average, the probability of having a positive NPV tends to increase over time. However, there are several drops in the curve, which show that optimizing for the best NPV and for the highest probability of having a positive NPV is different.

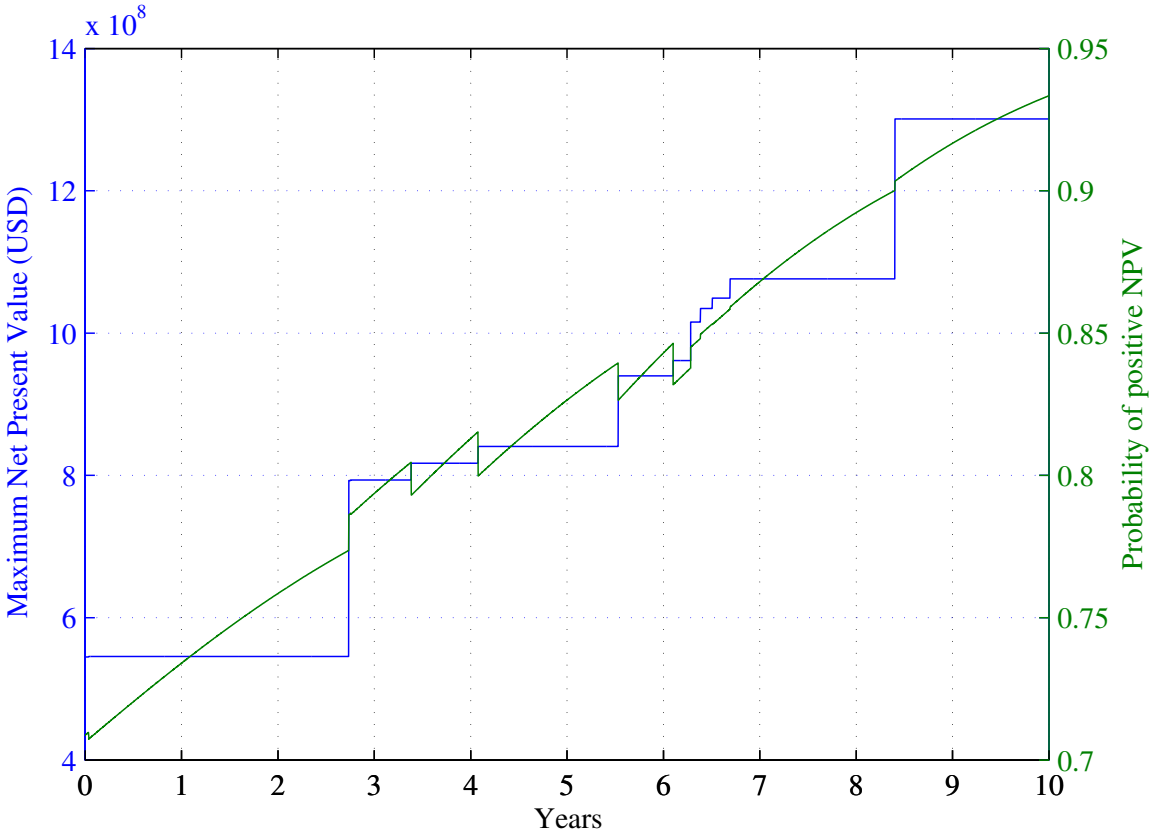



**Figure 157:** Probability of having a positive NPV for an optimum NPV

To evaluate this difference, Figure 158 presents the evolution of the highest probability of having a positive NPV while meeting the aforementioned constraint on the standard deviation of the NPV. Figure 158 shows that the optimum probability curve is monotonous with respect to time, as the degree of uncertainty in requirements decreases.

The analysis performed in this section is crucial to support decision makers in their trade-offs between risk and return when making go/no-go decisions. Hence, these results demonstrate the ability of the proposed methodology to quantify the benefits of waiting to start the program on the program profitability. In particular, the obtained results show that waiting to start the program seems to allow for better and more profitable concepts. However, this does not take into account the time value of money. Indeed, money now is worth more than money in the future. The consequence of this observation is discussed in the next section.

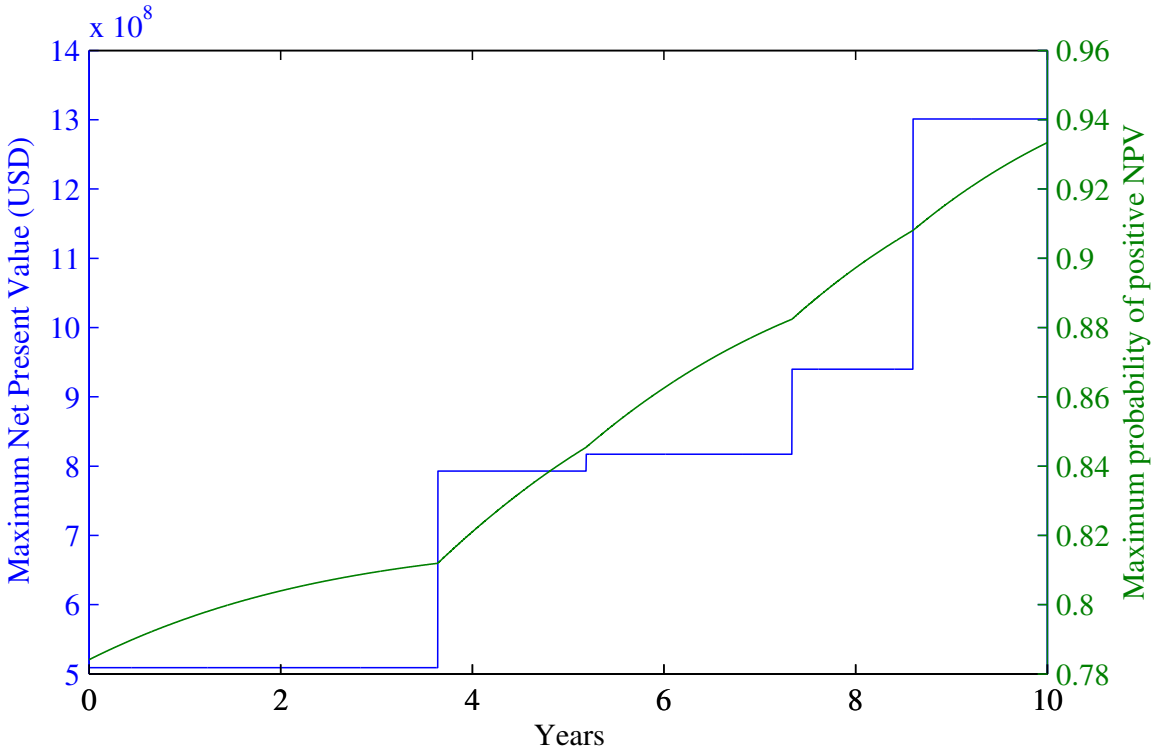



**Figure 158:** Optimum profit probability under a maximum $1 billion standard deviation

#### *9.3.7.2 Estimating the Best Program Start Date*

The goal of this section is to assess the capabilities of the proposed methodology to estimate the best start date of the program by trading potential returns with risks.

The effect of time on the objectives is ambivalent. On the one hand, waiting before starting a program leads to better defined requirements, and consequently less risk. On the other hand, delaying a program results in an opportunity cost, modeled by a reduction in NPV. Hence, time has two adverse effects that need to be captured and traded.

In order to perform such trade-offs, a simple scenario is studied. Using the set of optimal designs that was previously identified, an optimization is run. At each period of time within a timeframe of 10 years, the maximal feasible NPV is computed, while enforcing an arbitrary $1 billion standard deviation constraint on the NPV. Figure 159 shows the results of this analysis.

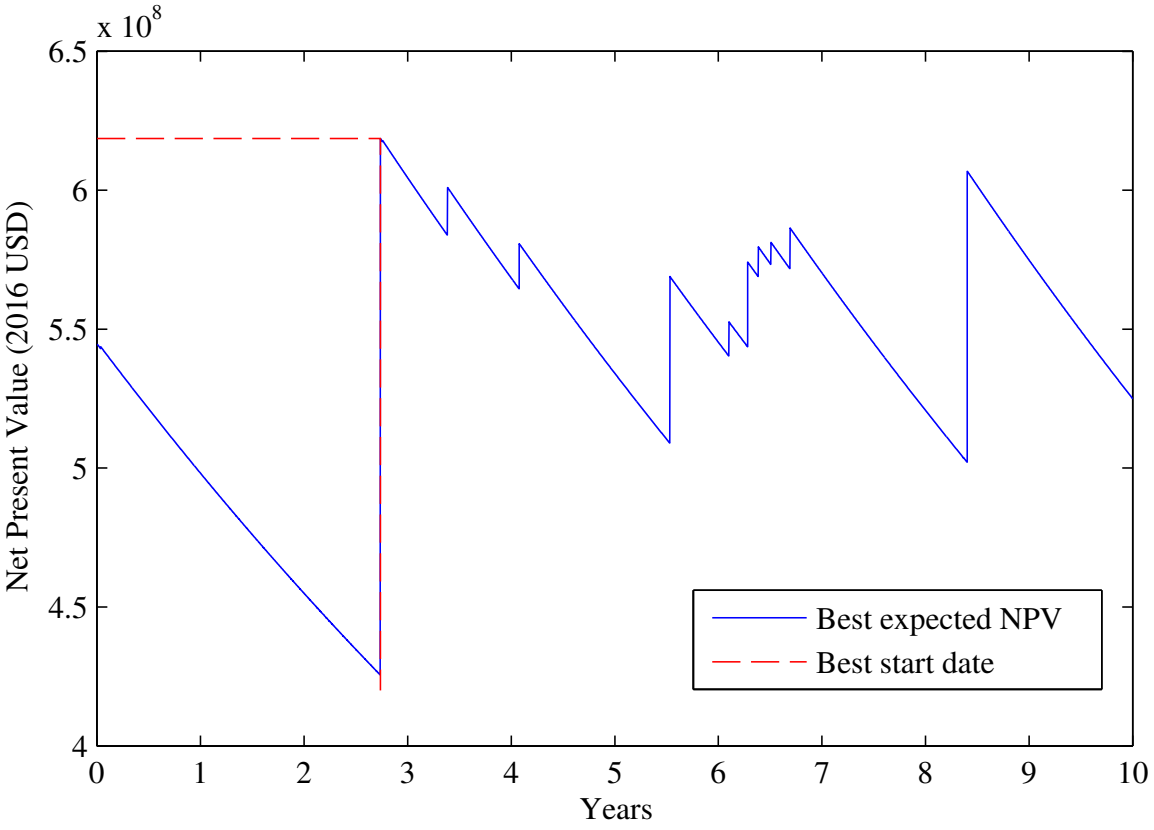



**Figure 159:** Evolution of the best expected NPV of future programs

As time goes by, the uncertainty associated with each concept progressively decreases, and concepts with greater profitability become feasible, which in turn results in a step in the largest feasible NPV. However, as time also reduces the NPV of each concept, it is not certain that the maximum feasible NPV would increase with time. In the case of this study, the maximum allowable NPV stems from a concept whose development should be started in a bit less than three years, when regulations will become more predictable. This optimum start date is highlighted by the red dashed line.

#### *9.3.7.3 Conducting Time-Dependent Risk and Return Analyses*

As mentioned in the previous section, the adverse behaviors of the risk (standard deviation of the NPV) and return (NPV) with time require trade-offs to be made. On the one hand, uncertainty decreases with time, as regulations and markets become more certain. On the other hand, delaying a program results in an opportunity cost intrinsically captured

by the NPV. This requires decision makers not only to select the best concept, but also to determine the best start date of the program. The previous section discussed this type of selection for a constraint representing an acceptable level of risk for the program. This idea is extended beyond this analysis to support the selection of the best set of expected NPV and NPV standard deviation for any program start time without constraint. In order to better understand the underlying behavior of the optimum NPV and standard deviation with time, Figure 160 displays the evolution of the Pareto frontiers for different values of the program start time.

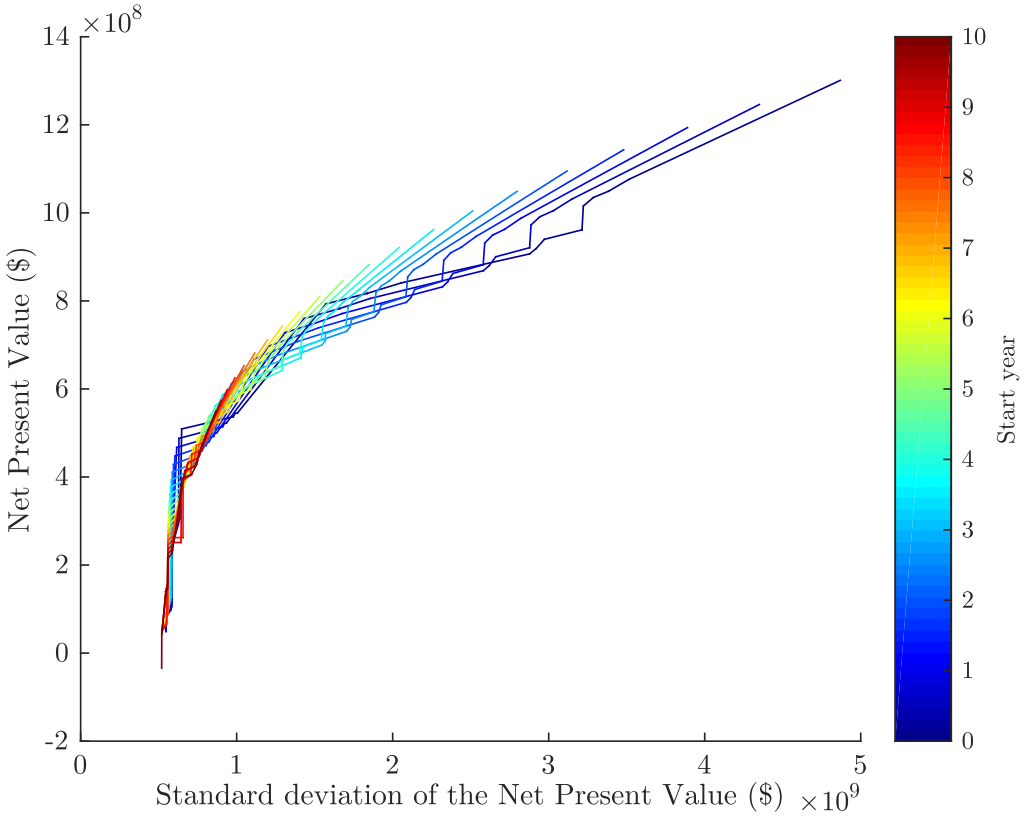



**Figure 160:** Pareto frontiers for different program start dates

As expected, waiting to start the program helps reduce its risk, but also limits the maximum potential profits. Indeed, as time goes by, the curves are shifted both to the left-hand side (smaller standard deviation) and to the bottom (smaller expected NPV). Delaying a program can result in a significant enough reduction in uncertainty, making it

relevant to consider the option of waiting.

In order to enable time-dependent trade-offs between risk and NPV, a single Pareto frontier is computed. All the curves generated in Figure 160 are combined and only the non-dominated points are kept. Figure 161 shows the result and provides the overall time-dependent Pareto frontier between Year 0 and Year 10.

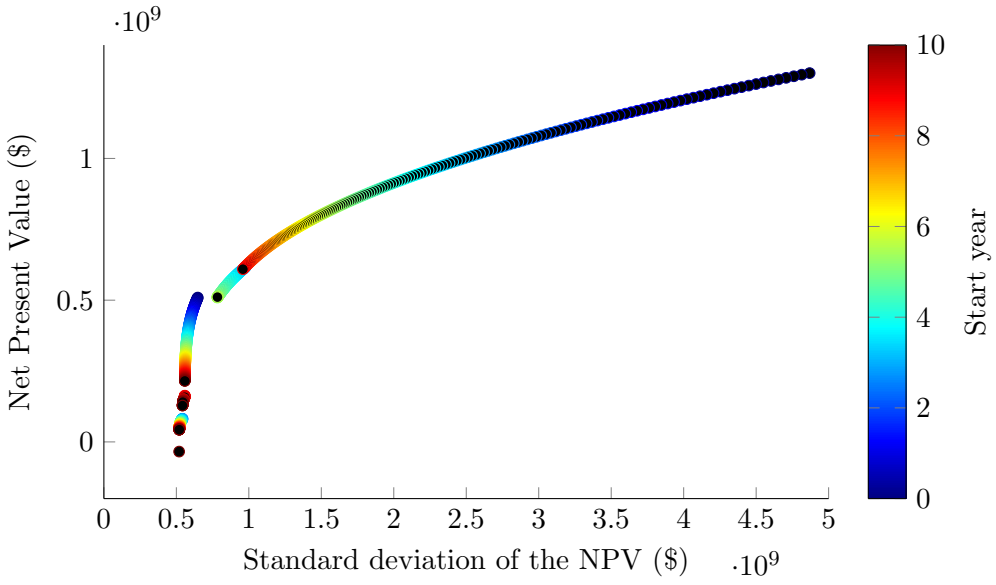



**Figure 161:** Time-dependent Pareto frontier of risk and return

The obtained Pareto frontier is fairly continuous, facilitating sound decision-making. Indeed, it provides decision makers with all the information required to make informed go/no-go decisions at any point in time. In particular, this analysis helps them trade risk against return while also supporting them in the selection of the best concept and start date of the program. As expected, concepts with a low standard deviation (left-hand side of the curve) tends to correspond to a late start date while concepts with a higher NPV tend to have an earlier start date. As decision makers agree on a maximum risk (standard deviation of the NPV) and a minimum return, Figure 161 helps them both optimize the start date of the program and find the corresponding optimum vehicle. As aforementioned, this analysis allows decision makers to find the vehicle corresponding to each point of the Pareto frontier. For that purpose, the selection methodology described in Section 7.3 is applied to find the best concepts for all possible combinations of relative importance given

to the NPV and its standard deviation.

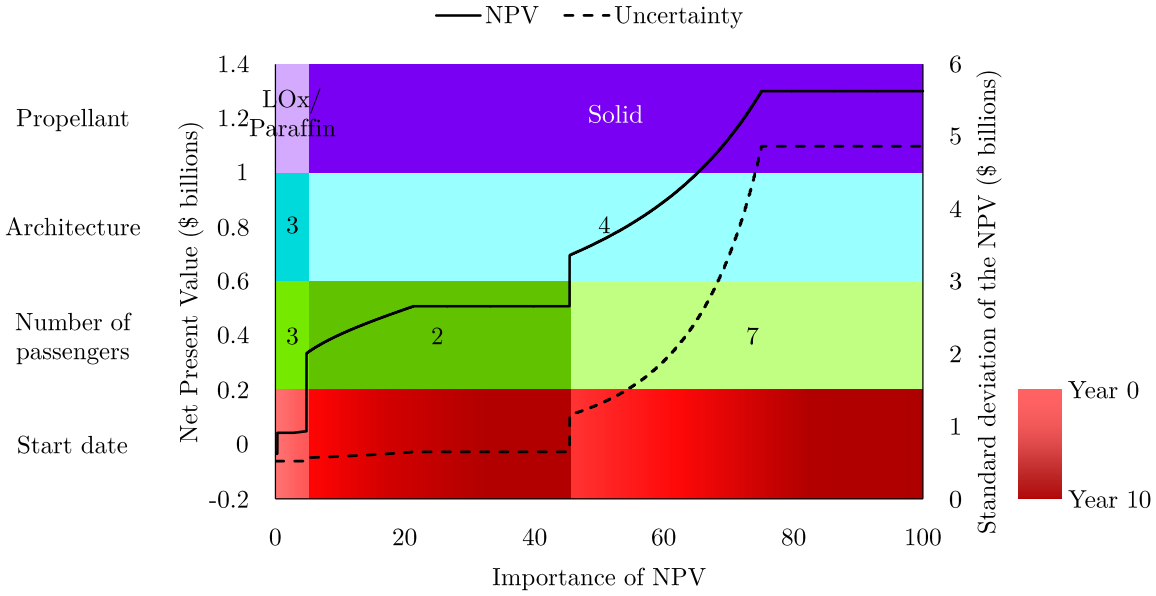



**Figure 162:** Identification of the baselines while trading performance against robustness

Figure 162 provides the main characteristics of such vehicles along with the start date. Given the importance of the NPV (x-axis) compared to its standard deviation, both the risk (standard deviation of the NPV) and the return (NPV) are displayed, along with the key vehicle parameters. Figure 162 demonstrates that the Pareto frontier is mostly composed of a few concepts with variable program start date. In other words, each concept represents a segment of the Pareto frontier, rather than a single point. It also clearly shows, as expected, that increasing the vehicle financial performance results in increasing the overall program risk. Finally, this analysis shows that smaller vehicles with two or three passengers could help alleviate the program risk by increasing the program robustness. Indeed, decreasing the vehicle size alleviates the uncertainty related to the demand.

To conclude, throughout this section, it was demonstrated that the proposed methodology has the capabilities required to support decision makers in making complex and risky go/no-go decisions in presence of evolving uncertainty in requirements. In particular, it allows them to both evaluate the best reachable performance under a maximum risk limit and estimate the best start date of the program, along with the corresponding optimum

vehicle. These observations lead to the validation of the last criterion:

**Validation Criterion 7: The proposed methodology is capable of generating the information required to support informed go/no-go decisions.**

## *9.4 Summary*

This section aims at summarizing both the characteristics and the capabilities of the proposed methodology when compared to other existing design space exploration methodologies. For that purpose, the Overarching Hypothesis is first validated, along with the two main Hypotheses established in Chapter 2. A parallel is then made between the steps of the different methodologies. Next, the capabilities are compared with respect to the criteria identified in Section 9.1. Finally, the new observations highlighted by the application of the proposed methodology to suborbital vehicles are presented.

### 9.4.1 Hypothesis Validation

The Overall Experiment described in Section 9.2 implicitly incorporates Experiments 1 and 2. Hence, this section aims at validating Hypotheses 1 and 2, as well as the Overarching Hypothesis.

#### *9.4.1.1 Hypothesis 1: Design Space Exploration*

As discussed in Section 2.1, validating Hypothesis 1 requires to ensure that the proposed methodology enables new promising concepts to be rapidly identified in a design space where alternatives are not necessarily defined by the same variables and/or evaluated with the same modeling and simulation environment. In this context, Section 9.3.1 demonstrates the ability of the proposed methodology to systematically optimize and compare alternatives that are not defined by the same design variables and/or the same modeling and simulation environment. In addition, Section 9.3.2 shows that the proposed methodology is more efficient than other approaches in providing a complete coverage of the design space. Finally, it has been demonstrated in Section 9.3.5 that the methodology has generated several innovative and promising alternatives that have not been found in the literature. For example, none of the existing concepts was built around a winged body powered by a

hybrid rocket engine and three jet engines. All these observations lead to the validation of Hypothesis 1.

**VALIDATION HYPOTHESIS 1: IF all feasible alternatives are systematically generated using a variable-oriented morphological analysis AND IF they are simultaneously compared and optimized using an evolutionary multi-architecture multi-objective algorithm based on architecture fitness THEN large design spaces can be better explored at a conceptual design level.**

#### *9.4.1.2 Hypothesis 2: Decision-Making Under Evolving Uncertainty in Requirements*

As discussed in Section 2.2, to validate Hypothesis 2, one must ensure that the proposed methodology is capable of both addressing evolving uncertainty in requirements and supporting critical decisions. For that purpose, Section 9.3.4 demonstrates the ability of the proposed methodology to perform important trade-offs that allow decision makers to identify key design drivers and key trends. Section 9.3.5 also shows that informed decisions are supported through the generation of well-described solutions that provide precious performance, design, financial, and program information. It has been shown in Section 9.3.6 that the proposed methodology is able to perform rapid trade-offs between performance and robustness, as well as forecasting the impact of decreasing uncertainty on these trade-offs. Finally, Section 9.3.7 demonstrates the ability of the methodology to help critical go/no-go decisions in the presence of evolving uncertainty in requirements. These observations result in the validation of Hypothesis 2.

**VALIDATION HYPOTHESIS 2: IF fuzzy set theory is used to propagate requirements' uncertainty whose magnitude has been modeled by scalable time-dependent membership functions AND IF a Design of Experiments of these scaling parameters is used to further create a TOPSIS THEN informed decisions can be made under fuzzy objectives and evolving uncertainty in requirements.**

#### *9.4.1.3 Overarching Hypothesis*

Finally, the successful evaluation of all validation criteria presented in Sections 9.3.1 to 9.3.7 results in the validation of the Overarching Hypothesis.

**VALIDATION OVERARCHING HYPOTHESIS: IF a variable-oriented morphological analysis is used to feed an evolutionary multi-objective multi-architecture optimization algorithm AND IF fuzzy set theory is used to parametrically propagate requirements' uncertainty through a multi-disciplinary physics-based modeling and simulation environment THEN large multi-architecture design spaces can be better explored AND informed decisions can be made under evolving uncertainty in requirements.**

### 9.4.2 Methodology Comparison

While most of the design approaches are built around the generic top-down design decision support process [67, 270, 384, 385], each of them has its specific methods to address the different steps. This section aims at comparing the founding pillars of the existing approaches with the ones of the proposed methodology, as displayed in Figure 163.

None of the existing approaches includes a method to efficiently model the degree of uncertainty in both the requirements and the design constraints. Most of the robustness analyses rely on Monte Carlo analyses based on uniform or normal uncertainty distributions. In addition, none of the existing approaches supports a systematic generation of all feasible alternatives, which can then be directly optimized and compared. In addition, there is a lack of approaches that enable decision makers to both conduct trade-off analyses and rank the concepts with respect to a given set of fuzzy objectives. Finally, only the proposed methodology generates the required data to allow decision makers to dynamically visualize all concepts of interest. As shown, the proposed methodology addresses more challenges of the design problem and provides deeper analysis capabilities. As a consequence, the proposed methodology offers a better approach for designing complex vehicles in both depth and breadth.

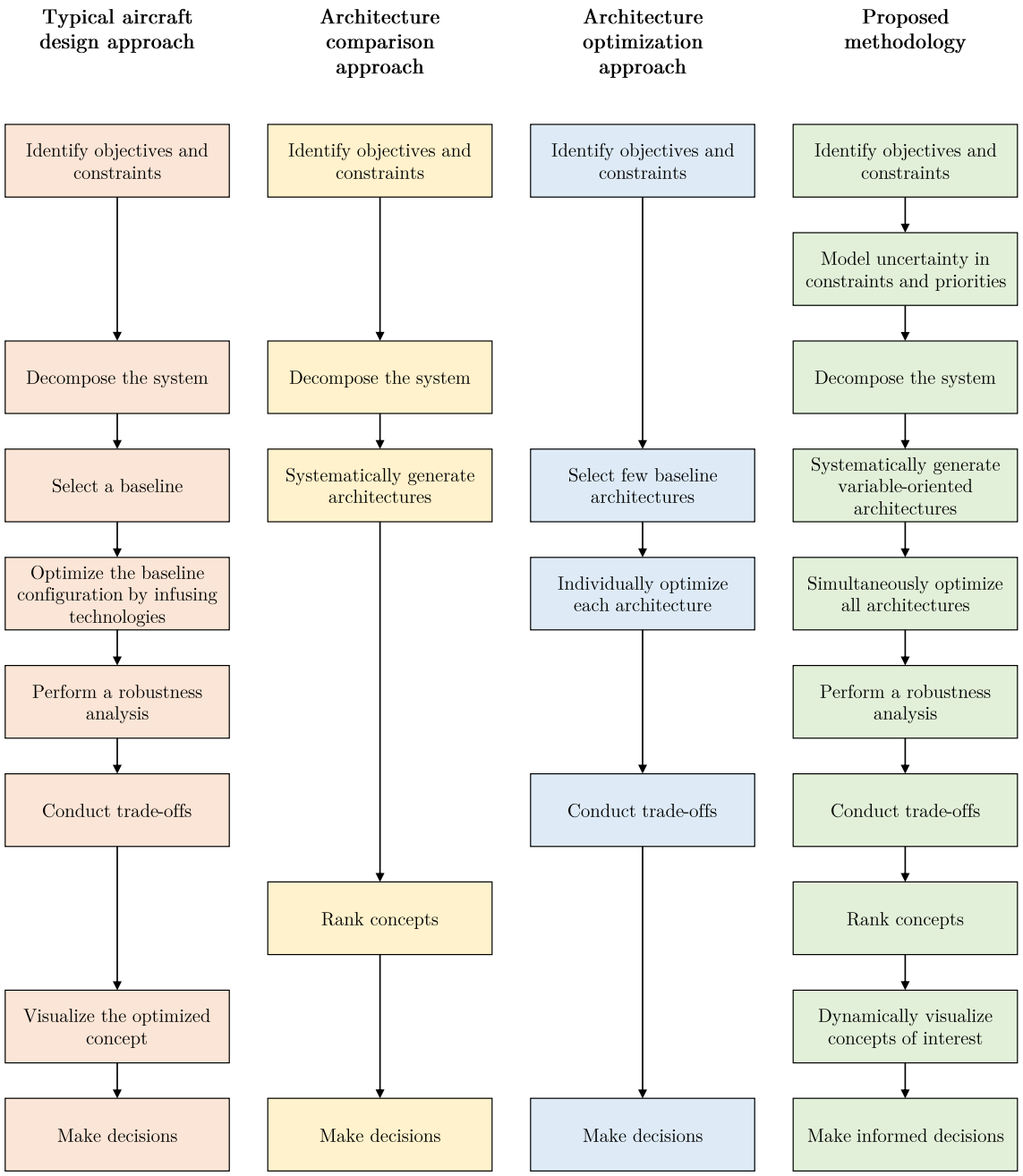



**Figure 163:** Comparison of the proposed methodology with existing approaches

### 9.4.3 Capability Comparison

This section benchmarks the capabilities of the different approaches with respect to the seven criteria identified in Section 9.1, as described below and summarized in Table 59.

- Criterion 1: optimizing and comparing different concepts with enough accuracy requires the design approach to be **flexible**, which means that it is able to evaluate alternatives with different modeling and simulation environments. The typical aircraft design approach is specific to a single baseline and cannot be extended to multiple design frameworks. The architecture comparison approach does not include a design framework as it is based on qualitative data. Hence, only the architecture optimization approach and the proposed methodology benefit from the required flexibility.

- Criterion 2: in order for the entire design space to be explored, the design approach must be **fast to run**. As the architecture comparison approach is based on qualitative data, it is computationally efficient. The typical aircraft design approach only analyzes one concept but it uses high-fidelity tools, which are extremely long to set up and run. All the other approaches have a medium computational efficiency as they have a lower level of detail than the typical aircraft design approach but evaluate more concepts.

- Criterion 3: avoiding the risk of missing any opportunity requires the approach to have a good **design space coverage**. The typical aircraft design approach only covers a tiny region of the design space located around the baseline. On the one hand, the architecture comparison approach generates points everywhere in the design space but lacks of quantitative analysis. On the other hand, while the architecture optimization approach alleviates the drawbacks of the two other methods, it only generates a handful of baselines so that the coverage is not exhaustive. Hence, the proposed methodology is the only one able to exhaustively cover the entire design space.

- Criterion 4: to support decision makers, it is crucial to help them conduct **trade-off analyses**. This requires quantifiable data, so the architecture comparison approach cannot be used. In addition, both the typical aircraft design approach and the architecture optimization approach do not cover enough concepts to support such analysis. The proposed methodology is the only one capable of generating quantifiable information about enough concepts to enable such trade-off analyses and trend identification.

- Criterion 5: to support informed decision-making, a **complete picture** must be provided. However, only the typical aircraft design approach and the proposed methodology generate enough data and are accurate enough to output a complete description of the optimized concept from both a physical and a business standpoints.

- Criterion 6: when a large amount of uncertainty is present in the requirements, there is a need to trade **performance against robustness**. This requires the uncertainty to be modeled and propagated, as partially done in the architecture optimization approach and the typical aircraft design approaches. The architecture comparison approach is not accurate enough to enable such analysis. The proposed methodology provides the modeling, propagation, and analysis capabilities to conduct such trade-offs.

- Criterion 7: in order to alleviate the program risk, there is a need for the approach to support informed **go/no-go decisions**. In particular, the evolving nature of requirements' uncertainty has to be considered and analyzed to help decision makers find the best start date of the program along with the corresponding vehicle. This type of analysis has not been conducted yet in any existing approach.

**Table 59:** Comparison of the capabilities of the different methodologies

| Criteria | Typical aircraft design approach | Architecture comparison approach | Architecture optimization approach | Proposed methodology |
|---|---|---|---|---|
| **Flexible** | | | ✓✓ | ✓✓ |
| **Fast to run** | ✓ | ✓✓ | ✓ | ✓ |
| **Design space coverage** | | ✓ | ✓ | ✓✓ |
| **Trade-off analyses** | ✓ | | ✓ | ✓✓ |
| **Complete picture** | ✓✓ | | ✓ | ✓✓ |
| **Performance vs. robustness** | ✓ | | ✓ | ✓✓ |
| **Go/no-go decisions** | | | | ✓✓ |

In order to visualize the capabilities of the proposed methodology against the other aforementioned methodologies, a radar plot is created and presented in Figure 164.

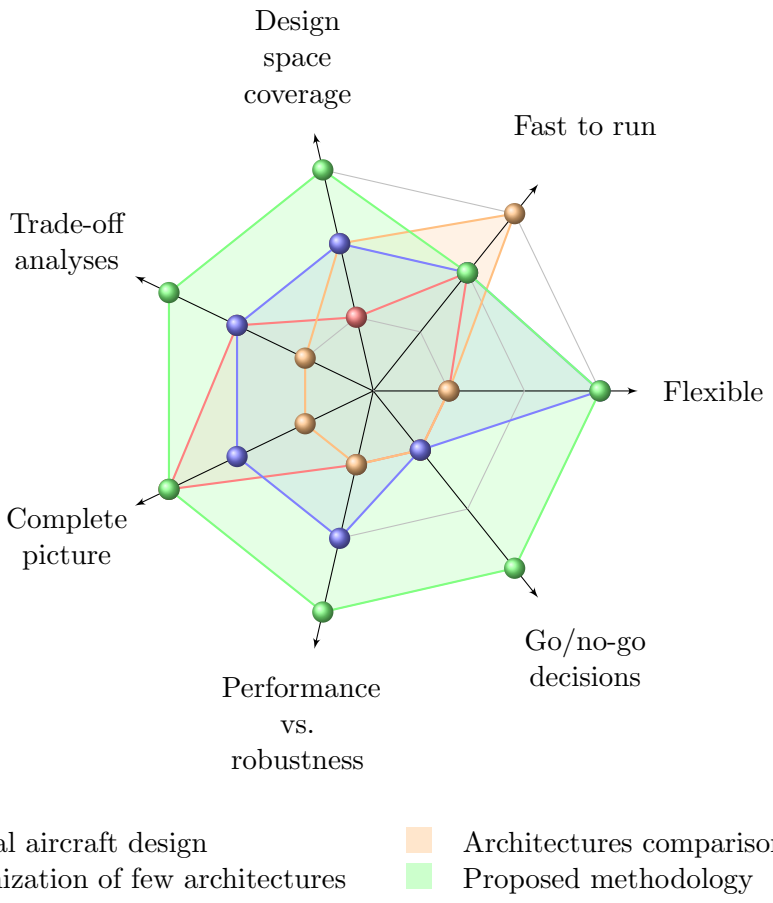



**Figure 164:** Capabilities of the proposed methodology compared to existing approaches

Applying the proposed methodology allows decision makers to make new crucial observations about emerging markets, as discussed in the next section.

### 9.4.4 New Observations

The proposed methodology has been applied to the design of a profitable commercial suborbital program and has provided new insights.

Indeed, the overall Pareto frontier provided in Figure 127 shows that the demand is strong enough for a suborbital program to be profitable. In particular, the maximum expected NPV that can be generated after 15 years is estimated at about $1.25 billion. In addition, no technology gaps that would prevent the success of such market has been

identified. Even though the most profitable concepts tend to be risky, there is still a cluster of relatively safe and profitable concepts. These concepts are built around Architectures 3 and 4, powered by a hybrid LOx/Paraffin engine, and carrying 6-7 passengers. The analysis demonstrates that, while being the safest, vertical take-off and vertical landing vehicles cannot be profitable for commercial suborbital tourism. The results also show that both Architecture 2 and liquid propellants are not promising for such applications and can be abandoned in favor of other types of technologies.

The robustness analysis demonstrates the importance of accounting for the program risk, especially at the dawn of this new market. Indeed, using the proposed methodology to select a robust concept would enable decision makers to increase the probability of making profit by around 10% if they start the design early. Besides, including the variance of the performance in the list of decision criteria enables to highly decrease the project risk by around 72%, while only reducing its performance by around 10%.

The application of the proposed methodology to support go/no-go decisions also shows that, if the uncertainty in requirements follows the assumed trends, the best start date of the commercial suborbital program would be in 2019. This date provides the best trade-off between opportunity and uncertainty-related risks.

Finally, a total of six concepts of interest have been identified as being potential baselines for further detailed analyses. The performance-focused analysis highlights four dominant concepts depending on the importance given to either safety or NPV. Accounting for either robustness or passenger experience leads to two additional promising concepts.

As a consequence, the proposed methodology provides key insights on the future suborbital market. The previous analysis also demonstrates the additional value captured by the proposed methodology when compared to traditional design space exploration approaches.

# CHAPTER X

# CONCLUSION

## *10.1 Summary of the Research*

Recent technological developments have resulted in the emergence of innovative and complex advanced vehicles such as flying cars, suborbital vehicles, and hypersonic aircraft. As discussed in Chapter 1, there are many challenges associated with the development of the markets opened up by these new vehicles. In particular, these new markets are characterized by a large multi-objective decision environment, where flying performance is competing against economical viability, passenger safety, program risk, etc. The impact of such objectives on the program's ability to enter a new market requires economic, safety, and robustness metrics to be considered in early design phases. In addition, the complexity of these new vehicles gives rise to a large combinatorial space of possible configurations for which no baseline has been established and experts' judgment is highly limited by the lack of experience. A successful market penetration requires designers to define an optimized baseline and to identify both the main design drivers and potential technology gaps. Another major challenge is the presence of evolving uncertainty in requirements due to the lack of experience and established regulations. Hence, flexible decision-making techniques are needed to alleviate risks inherent to the launch of new programs. This research aims at supporting the development of emerging markets by establishing a methodology that enables a broad design space exploration at a conceptual level to select solutions against unclear objectives and under evolving uncertainty in requirements.

A review of current design approaches in Chapter 2 highlighted a lack of efficient design space exploration techniques. Current methods are indeed only capable of either comparing, at a high-level, numerous architectures or optimizing a handful of alternatives with respect to more detailed parameters. Moreover, while numerous robust design techniques have

been developed in the literature, none has been found efficient enough to support informed decisions under time-dependent uncertainty in requirements.

To bridge these gaps, a four-step methodology described in Chapter 3 and further detailed in Chapters 4 to 7 is developed. The founding pillars of this methodology can be directly mapped to the ones commonly used in the generic top-down design decision support process. Hence, while the approach followed in this research is generic, the specificity of the proposed methodology and its unique capabilities rely on the methods implemented to address each step. To demonstrate these capabilities, a proof-of-concept was developed around the design of a profitable commercial suborbital program.

In Step 1 (establish the decision criteria), decision criteria are established that both limit the available design space and are used to evaluate and compare the various design alternatives. This step consists in mapping customer requirements to quantifiable and precise metrics. Design objectives are clearly defined, along with their evaluation methods. In a multi-objective solution space, the relative importance given to each objective is modeled, as well as its level of confidence. The design constraints that limit the design space are also modeled using time-dependent membership functions. In particular, each constraint is defined by a mean value, a standard deviation, and the evolution of these parameters over time.

In Step 2 (define the design space), alternatives covering the entire design space are generated to be further optimized and compared. For that purpose, a new variable-oriented morphological analysis combined with a compatibility analysis has been developed and implemented into a generic tool named ENVISAGE. This method is built around a five-step process, as described below:

1. Decompose the system into functions using a tree diagram: starting from the high-level goal, functions are identified using a top-down functional breakdown.

2. Generate all possible combinations: a morphological matrix is used to list all possible options for each function. This matrix is a two-dimensional representation of the

system, where each row represents a function/feature of the system and each column represents an option.

3. Ensure feasibility via a compatibility matrix: this square symmetric matrix is used to define compatibility between each couple of options.

4. Define design variables for each function and assign them to their corresponding option(s).

5. Generate feasible architectures by grouping alternatives that are described by the same variables.

When applied to suborbital vehicles, this method demonstrates great capabilities. Indeed, starting with around 900 million possible combinations, the method identified 47 million feasible concepts grouped into four architectures. While all existing concepts are included in the list of feasible generated alternatives, the number of discrete optimization problems that have to be executed are reduced by several hundred thousands. Moreover, the concepts can be further optimized as concepts described by the same design variables are grouped into well-described variable-oriented architectures.

In Step 3 (evaluate alternatives), the alternatives identified have to be systematically evaluated in order to support quantitative analyses and trade-offs. A thorough literature review revealed a lack of readily available modeling and simulation environments that enable an accurate and rapid evaluation of flying, economic, and safety performance of all types of suborbital vehicles at a conceptual design level. To bridge this gap, a sizing and synthesis environment based on a modified MDF approach and using both empirical relations and surrogate models was developed. This environment consists in six disciplinary modules: weight/size, aerodynamics, trajectory, propulsion, economics, and safety. In addition, 26 design variables are required to define a concept. The environment was validated using three existing concepts (New Shepard, RocketPlan XP, and SpaceShipTwo). The results were deemed accurate for a conceptual design level as the mean error is around 15% for both size and weight. In addition, the environment only takes about ten seconds to run

and is capable of considering all types of suborbital vehicle configurations. Finally, it provides all the required information: flying, economic, and safety performance metrics. When developing this design framework, the main challenge to be overcome was the modeling of all types of chemical rocket engines. As such, a design methodology was presented that supports the design of future rocket-powered aerospace vehicles. In particular, it provides the capabilities to rapidly evaluate the performance, weight, size, and life-cycle costs of all chemical rocket engines at a conceptual level.

In Step 4 (make informed decisions), a decision-making process has been developed. A literature review demonstrated a lack of techniques able to systematically optimize and compare alternatives that are defined by different sets of design variables and/or by different modeling and simulation environments. For that purpose, a new evolutionary multi-architecture multi-objective optimization algorithm called EMMA was developed. It consists in a four-step process:

1. Individually optimize each architecture using specifically configured NSGA-II with an initial number of generations common to all architectures.

2. Evaluate the performance of each architecture compared to others based on their location on the overall Pareto frontier.

3. Re-execute each NSGA-II with a specific number of generations calculated with an evolutionary algorithm as a function of the performance of its corresponding architecture.

4. Repeat steps 2 and 3 until the convergence criterion is met.

This approach has been validated on multiple test functions and provide significant improvements when determining complex multi-architecture Pareto frontiers.

The goal of the decision-making environment is not only to provide rapid trade-off analysis capabilities but also to enable a systematic and rigorous selection of the best concept(s) with respect to a given set of design objectives. This is achieved by implementing

a MADM technique to rank the different concepts according to the stated criteria. In particular, the TOPSIS appeared to be the most suitable technique for this research. In order to account for uncertainty in design priorities, a three-step approach to select the most robust design(s) was developed. First, uncertainty is defined using probability distributions in place of single values to represent the multiple design priorities. In particular, Gaussian distributions, characterized by both the most probable value and the standard deviation, are used. Second, uncertainty is propagated using a Monte Carlo simulation. It samples the previously modeled Gaussian distributions for each variable in order to produce a large number of possible sets of priorities, also called scenarios. For each scenario, alternatives are ranked using the aforementioned TOPSIS. Third, the concept selection is performed by outputting a final ranking. The latter is calculated by averaging the rank of each alternative over all generated scenarios. This approach provides a probabilistic way to model, propagate, and capture uncertainty in design priorities. It also supports the selection of concepts that are both optimally performant and robust to changes in design priorities.

When dealing with emerging markets, a large amount of uncertainty is present in requirements. Hence, there is a need to include robustness in the decision-making process. For that purpose, fuzzy set theory is used to model and propagate time-dependent uncertainty throughout the design framework.

Finally, a parametric and dynamic visualization platform is developed to both bridge the gap between the information provided by the optimization algorithm and the human cognitive and perceptual systems and facilitate the integration of designers' past experience, knowledge, and cognitive capabilities in the analysis. It also supports a more traceable and faster decision-making process.

Through its application to the design of a profitable commercial suborbital program, this methodology has demonstrated that it provides the seven key features needed to support the exploration of large design spaces in presence of evolving uncertainty in requirements. First, alternatives evaluated with different modeling and simulation environments can be systematically compared and optimized. Second, the computational time of a complete

design space exploration is decreased while also increasing the number of concepts investigated. Third, the region of the design space covered is larger than current approaches. Fourth, quantitative trade-off analyses can be performed and key trends can be identified among all alternatives. Fifth, informed decision-making is supported by a complete picture of the selected concepts from both the physical and the business standpoints. Sixth, the program risk can be easily alleviated by the ability to trade performance against robustness in early design phases. Finally, all the information required to support critical go/no-go decisions can be generated.

Ensuring that the proposed methodology benefits from all the aforementioned capabilities results in the validation of the overarching research hypothesis: **if a variable-oriented morphological analysis is used to feed an evolutionary multi-objective multi-architecture optimization algorithm and if fuzzy set theory is used to parametrically propagate requirements' uncertainty through a multi-disciplinary physics-based modeling and simulation environment then large multi-architecture design spaces can be better explored and informed decisions can be made under evolving uncertainty in requirements.**

The development and the application of the proposed methodology result in the numerous key contributions, as discussed in the following section.

## *10.2 Research Contributions*

The methodology developed in this research, named ASCEND, is generic and can be applied to any kind of problems characterized by a large multi-architecture design space and in presence of evolving uncertainty in requirements. To demonstrate its capabilities, it has been applied to suborbital vehicles. The work presented herein provides multiple contributions in several domains: design space exploration, decision-making under evolving uncertainty in requirements, and conceptual design of suborbital vehicles.

### 10.2.1 Design Space Exploration

Current approaches are limited by their inability to systematically optimize and compare alternatives that are not defined by the same design variables and/or modeled by the

same design framework. Either the quantity of considered alternatives or the quality of the optimization is lacking. As a consequence, these approaches cannot support the development of emerging markets opened by new advanced vehicles. The proposed methodology overcomes this flaw by improving both alternative generation and alternative selection processes. Indeed, the proposed variable-oriented morphological analysis is capable of including design variables in the concept generation algorithm. Hence, the number of distinct architectures generated is greatly reduced while the number of possible alternatives considered is implicitly increased. This novel methodology has been implemented into ENVISAGE, which is a generic user-friendly software developed with Matlab. The application of the proposed approach to suborbital vehicles demonstrated great benefits. Indeed, the number of discrete optimization problems to be executed is reduced by a factor of $10^5$. This results in crucial improvements in the number of function calls, when used to explore large design spaces.

To simultaneously compare and optimize these architectures, a new evolutionary multi-objective multi-architecture optimization algorithm named EMMA is proposed based on the notion of architecture fitness. EMMA provides designers with the ability to efficiently, systematically, and rigorously optimize concepts that are not described by the same design variables and/or evaluated with the same modeling and simulation environment. In addition, EMMA is capable of generating more accurate and better sampled Pareto frontiers than existing design space exploration techniques.

As a result, the risk of missing promising concepts is greatly reduced compared to conventional architecture optimization methods. Similarly, the promising highlighted architectures are described more precisely and more quantitatively, leading to decisions that are less subjective and better documented. The decision traceability is also strongly improved. Finally, another benefit of this quantification is the resulting capability to identify the key design drivers and to assess the sensitivity of each parameter to the different objectives. When compared with existing approaches, the proposed methodology allows decision makers to find solutions 40% more performant for the same execution time or 40 times faster for the same accuracy.

### 10.2.2 Decision-Making Under Evolving Uncertainty in Requirements

Conventional robust design methodologies usually provide static capabilities to model uncertainty in requirements. Either distributions are assigned around most probable values or requirements' values are varying within a given range. However, these approaches do not match the dynamic behavior of uncertainty in emerging markets. Indeed, both the degree of uncertainty and the mean values are subject to changes throughout the establishment of the market. Hence, there is a need for more flexible capabilities, as the ones developed in this research, that better match this evolving behavior of uncertainty. In particular, fuzzy set theory enables the modeling and propagation of time-dependent uncertainty in requirements. As a result, the risk of selecting a design at different points in time can be assessed. In particular, the trade-off between waiting for more precise requirements and starting the detailed design can be quantified.

By quantifying the trade-off between risk and expected performance, this methodology also helps designers make challenging go/no-go decisions and provides them with the best start date of a program. In particular, in the context of suborbital vehicles, it provides a robust solution that increases the probability of success by 10% compared to the ones generated by traditional approaches.

Hence, scenarios can be developed in order to find how much should be known about requirements to ensure the success of a program. In addition, new questions can be addressed:

- Is it worth to design a single vehicle for two slightly different applications? What are the characteristics of such vehicle?
- Which uncertain requirements have the highest impact on the design?
- Which requirements must be addressed first to strongly narrow down possible configurations?
- When should the program be started to maximize the expected return while minimizing the risk?

ASCEND also allows designers to update objectives and requirements to match available market information. Thus, market analysis can be more efficiently included in the design process. In addition, the methodology helps decision makers communicate about risk and performance trade-offs with customers and regulatory entities. Finally, the visualization capabilities developed in the context of this research greatly facilitate collaboration among multi-disciplinary teams and present great marketing capabilities.

### 10.2.3 Conceptual Design of Suborbital Vehicles

The application of this methodology to suborbital vehicles requires new evaluation capabilities in terms of flying, economic, and safety performance. For that purpose, a new modeling and simulation environment has been developed around six disciplinary modules: weight/size, aerodynamics, trajectory, propulsion, economics, and safety. Nevertheless, the two major challenges were the development of the propulsion and the safety modules as they required capabilities that have not been found in the literature yet.

First, a new framework for performance, weight, and life-cycle cost estimation of rocket engines at a conceptual level has been developed. By leveraging cycle-based approaches and surrogate modeling techniques, the performance of all chemical rocket engines can be evaluated with an accuracy of 3%, while dividing the execution time by a factor of $10^5$ compared to current physics-based models. New mass estimating relationships have been developed for estimating the weight and size of solid engines with an improved accuracy compared to existing models. In addition, physics-based models built around the key design drivers are used for the weight and size estimation of liquid and hybrid engines. While existing cost estimating relationships are used to evaluate the life-cycle costs of solid and liquid engines, a more physics-based model was developed for hybrid engines. While supporting complex multi-objective optimization and rapid trade-off analysis, this environment is also the first of its sort able to estimate the life-cycle costs of hybrid rocket engines. Second, to overcome the lack of safety evaluation technique, a new method has been developed to enable a quantitative assessment of the risk of catastrophic failure. Finally, the integration of all the disciplinary modules enables the development of a new integrated modeling and

simulation environment more flexible than existing framework as it is the first of its sort able to accurately estimate the flying, economic, and safety performance of all types of suborbital vehicles at a conceptual design level.

The application of the proposed methodology to suborbital vehicles shows that a commercial suborbital program might be profitable if well designed. Among the most promising vehicles, air launched vehicles powered by solid engines appear to be the most profitable, yet risky. Horizontal take-off and horizontal landing vehicles powered by both a hybrid rocket engine and jet engines seem to be the most robust type of concepts. Finally, vertical take-off and landing vehicles powered by hybrid rocket engines are not profitable but seem to be the safest. Based on assumed requirements' uncertainty models, the proposed methodology suggests an optimum start date of the suborbital program in 2019.

## *10.3 Current Limitations and Recommendations for Future Work*

As the proposed methodology is intended to be generic, the next step would be to assess its capabilities using other design problems such as flying cars, hypersonic commercial aircraft, delivery drones, etc. For that purpose, the methodology could be implemented into a software that provides designers with a user interface and the computational efficiency required to tackle larger problems with more accurate modeling and simulation environments. In addition, for more accurate results, interfaces between systems and interfaces between the system of systems and human operators must be taken into account when decomposing the problem. Indeed, such considerations might highly impact the life-cycle costs, safety, and performance of the designed vehicles.

For validation purposes, the methodology has been applied to suborbital vehicles and a medium-fidelity modeling and simulation framework was developed in Matlab. However, the methodology could be linked to existing design frameworks that enable either a higher fidelity analysis of the proposed concepts or a faster and even broader design space exploration. A problem that requires multiple design frameworks could also be implemented to better demonstrate the flexibility of the proposed methodology. Besides, operational functions can be included in the functional decomposition in order to reduce the probability of

missing risks related to the vehicle operations.

While this research only focuses on the integration of multiple architectures, it can be extended to the integration of multi-fidelity tools. Indeed, instead of driving the optimization of multiple architectures based on the architecture fitness, the algorithm could be modified to drive environments with different levels of fidelity based on the fitness of the previous analysis. This would not only favor the optimization of promising concepts but also increase their level of fidelity.

While the modeling and simulation environment has been developed in Matlab, its implementation into a more efficient language would enable both more accurate results and more interesting studies. In particular, the time-dependent part of the uncertainty has only been applied to its standard deviation. However, the mean value of the uncertainty membership functions might also change. Consequently, multiple scenarios could be analyzed by changing both the degree of uncertainty and the most probable value of each requirement.

Finally, if a more accurate description of the vehicle is available, the methodology could be extended downstream with techniques such as 3D meshing and CFD analyses. For some concepts, rapid prototyping could also be included within the methodology.

# APPENDIX A

# DEVELOPMENT OF ENVISAGE

## *A.1 Development of the EfficieNt Variable-orIented Software for Architecture GEneration (ENVISAGE)*

The general structure of the tool is based on the method described in the previous section. First, the users input their morphological matrix based on a functional decomposition. Then, to ensure the feasibility of the generated concept, they complete a pre-filled compatibility matrix. After this point, two analysis options are proposed:

- Alternative-based analysis: users can generate the list of all feasible alternatives.
- Architecture-based analysis: after defining the design variables specific to each function and option, users can generate the list of all feasible architectures.

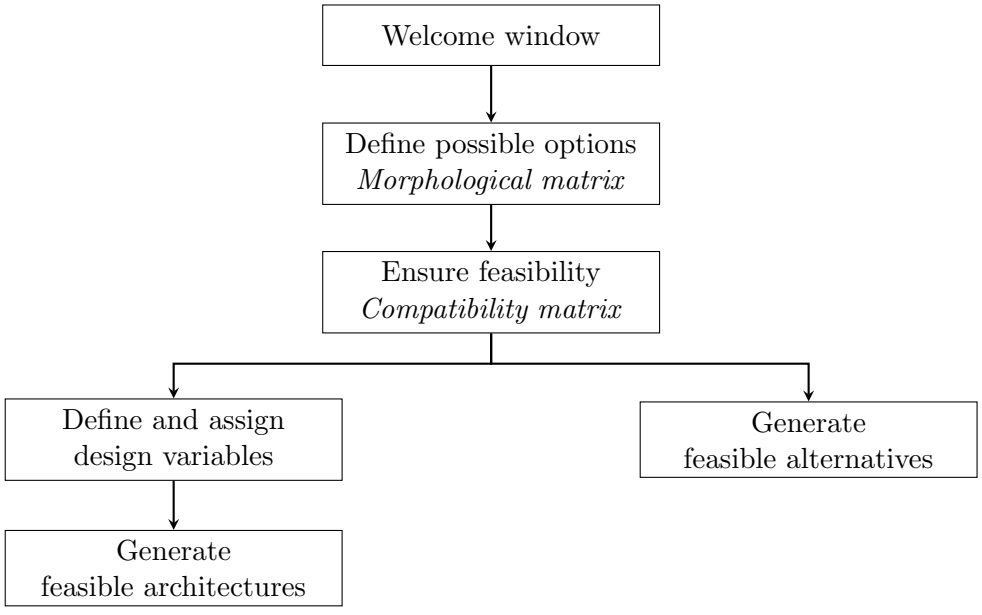



**Figure 165:** General architecture of ENVISAGE

While the general architecture of the tool is displayed in Figure 165, this section describes the main functions of the software. A notional case has been used to demonstrate

the capabilities of the software and this section refers to matlab functions presented in Appendix D.1.

### A.1.1 Welcome Window

The welcome window (`welcome.m`), displayed in Figure 166 presents the software and its author. It allows the user to start the program through the button "Start".


**Figure 166:** Welcome window of ENVISAGE

### A.1.2 Define Possible Options

This function is fulfilled by the matlab function `morphologicalMatrix.m` displayed in Figure 167. It allows users to build their morphological matrix. A first 2×2 matrix is presented with predefined feature names. Users can add features and options using the corresponding buttons. They can also change the name of each feature using the pop-up menu in the top right-hand corner. The selection of one of the features in the pop-up menu opens a small window `changeFeatureName.m` also displayed in Figure 167. Through this window, the users can change the name of the different features. To populate the matrix, users input the different options in the displayed table.

An option also allows users to upload a morphological matrix using an Excel file and to modify it. In addition, they can save their morphological matrix into an Excel file using the "Save" button.

Once the morphological matrix has been fully defined, the "Ensure compatibility" button allows users to define the compatibility between each option. If this button is pressed while the morphological matrix is empty, an error window (`noMorpho.m`) is displayed.

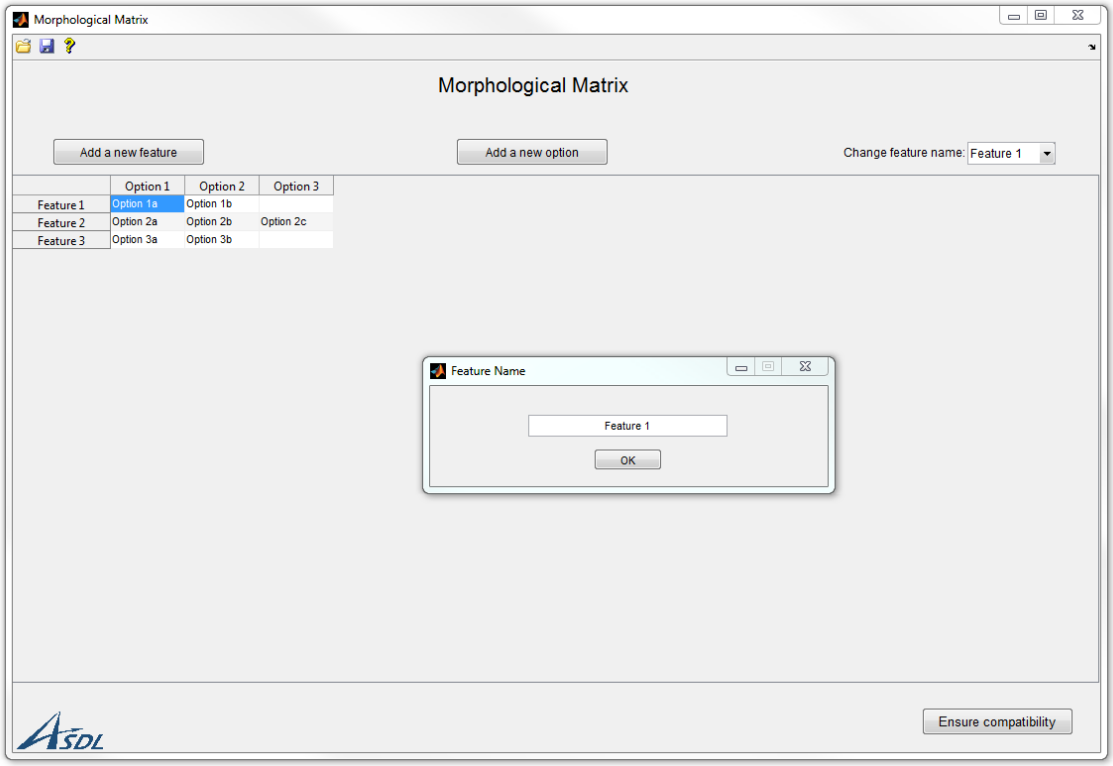



**Figure 167:** Morphological matrix of ENVISAGE

### A.1.3 Ensure Feasibility

To only generate feasible alternatives, the compatibility between each option must be defined through the function `compatibility.m`. Hence, based on the previously defined morphological matrix, a square compatibility matrix is automatically created. The names of all the features defined in the previous window are automatically reported in the new matrix. Since the compatibility matrix is symmetric, only the upper triangular is considered in this tool (Figure 168).

In addition, the compatibility matrix is automatically prefilled using the function `PreFillCompa.m` to account for the fact that two options of the same feature are necessarily incompatible. Users can load a predefined matrix and save the modifications to this matrix. If the loaded compatibility matrix does not have the correct dimensions, an error is displayed through the window `wrongCompat.m`.

Once the compatibility has been defined, two different analyses are proposed. The first

one corresponds to the traditional approach and lists all feasible alternatives. The second one allows users to define design variables and generate architectures based on the new approach.

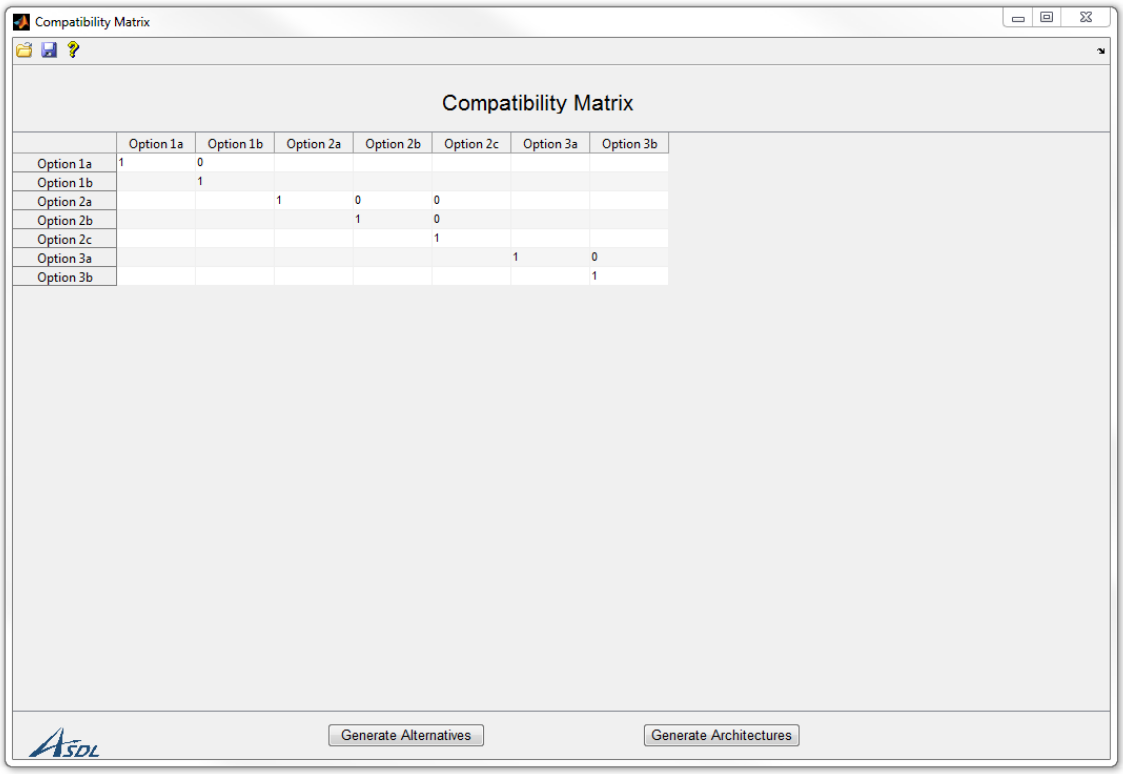



**Figure 168:** Compatibility matrix of ENVISAGE

### A.1.4 Generate Feasible Alternatives

The list of feasible alternatives (Figure 169) is displayed in the window `Alternatives-Display.m`. This window calls the main function `GenerateFeasibleAlternatives.m`, which is in charge of generating all feasible alternatives based on both the morphological matrix and the compatibility matrix. While this function uses `GenerateCompatibility.m` to convert the morphological matrix into a list of alternatives, it mainly relies on the function `AlgoCompa.m`. Indeed, the latter is a recursive function that searches for feasible combinations of options. This recursive function calls `CheckCompat.m` in order to check the compatibility between an option and all other options from a given list.

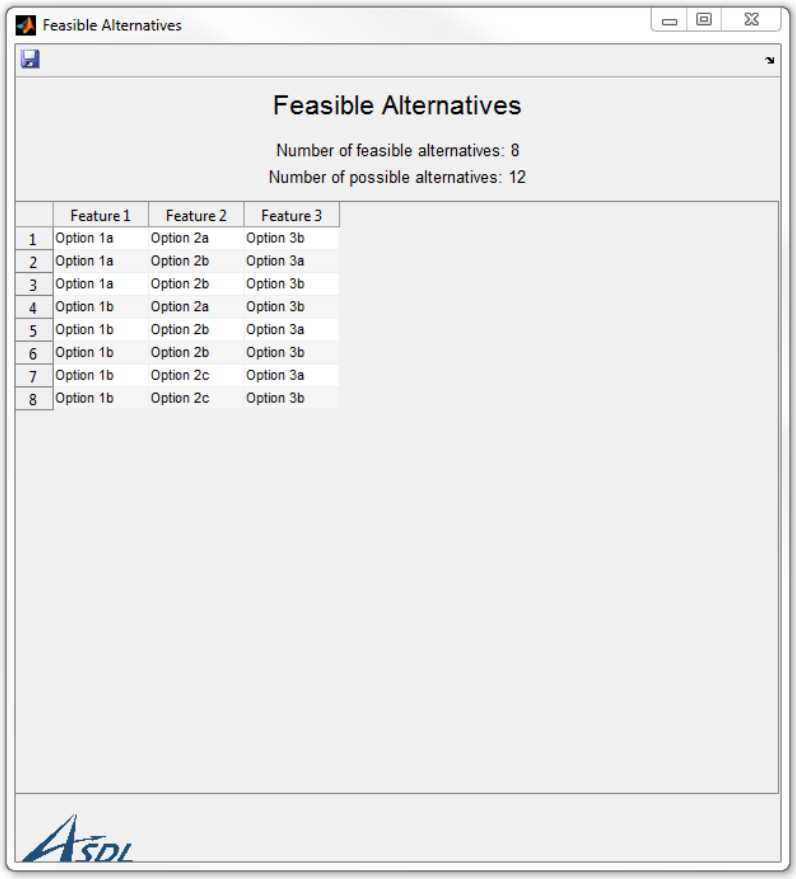



**Figure 169:** List of feasible alternatives of ENVISAGE

### A.1.5 Define and Assign Design Variables

The function `architectureDefinition.m` allows users to define the elements required to generate architectures. It displays the initial morphological matrix and allows users to choose the features they want to characterize through a pop-up menu. Once all features have been fully defined, the button "Generate Architectures" allows users to see the list of architectures.

Once a feature has been selected from the previous window, the window `defineFeatures.m` allows users to define the feature (Figure 171). Users can add variables and a description of each variable through the panel "Variables". Variables are then added to the table on the left-hand side and users can allocate the variables to the corresponding options

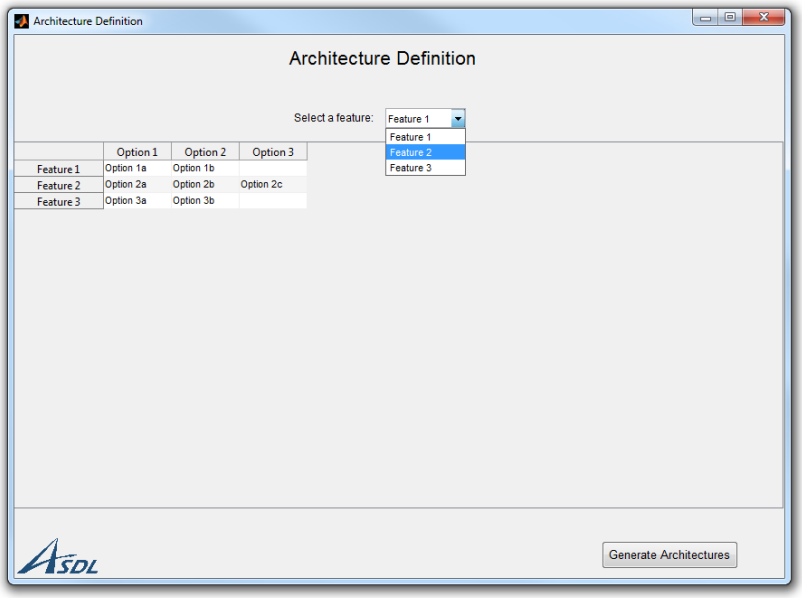



**Figure 170:** Architecture definition window of ENVISAGE

through check-boxes. The toolbar also allows users to save and load the variables/description table.

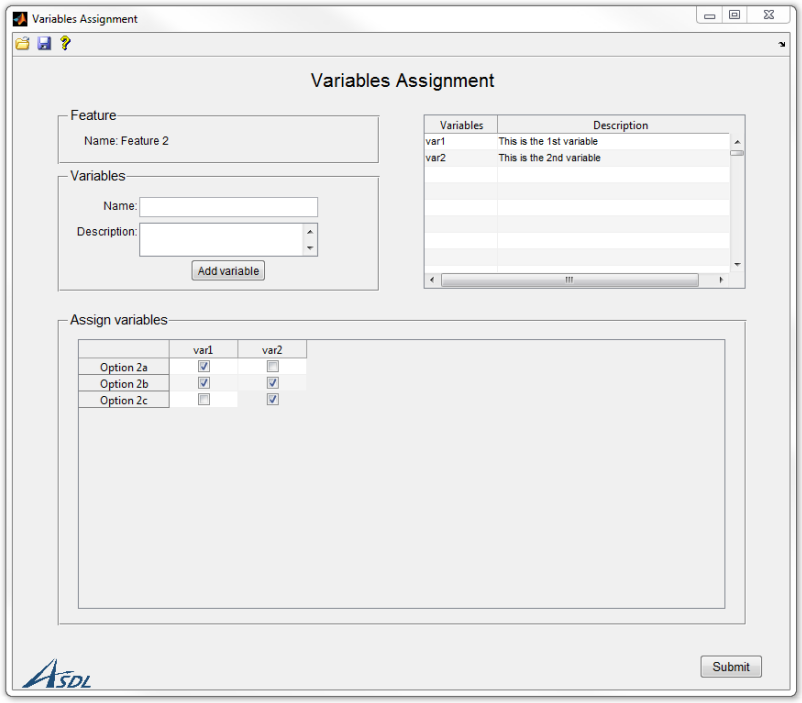



**Figure 171:** Variable assignment window of ENVISAGE

### A.1.6 Generate Feasible Architectures

The window `ArchitecturesDisplay.m` displays the list of feasible architectures. Similarly to `AlternativesDisplay.m`, it first determines the list of feasible alternatives. Then, the function `detectArchitectures.m` re-orders the aforementioned list in order to group alternatives that are defined by the same variables. Hence, the attributed variables and their description are needed and are provided by the function `GenerateMorphoVar.m` and `GenerateMorphoVar2.m`. Users can save the final table through the toolbar and can display the list of all variables along with their description through the button "Variable description". The window is displayed in Figure 172.

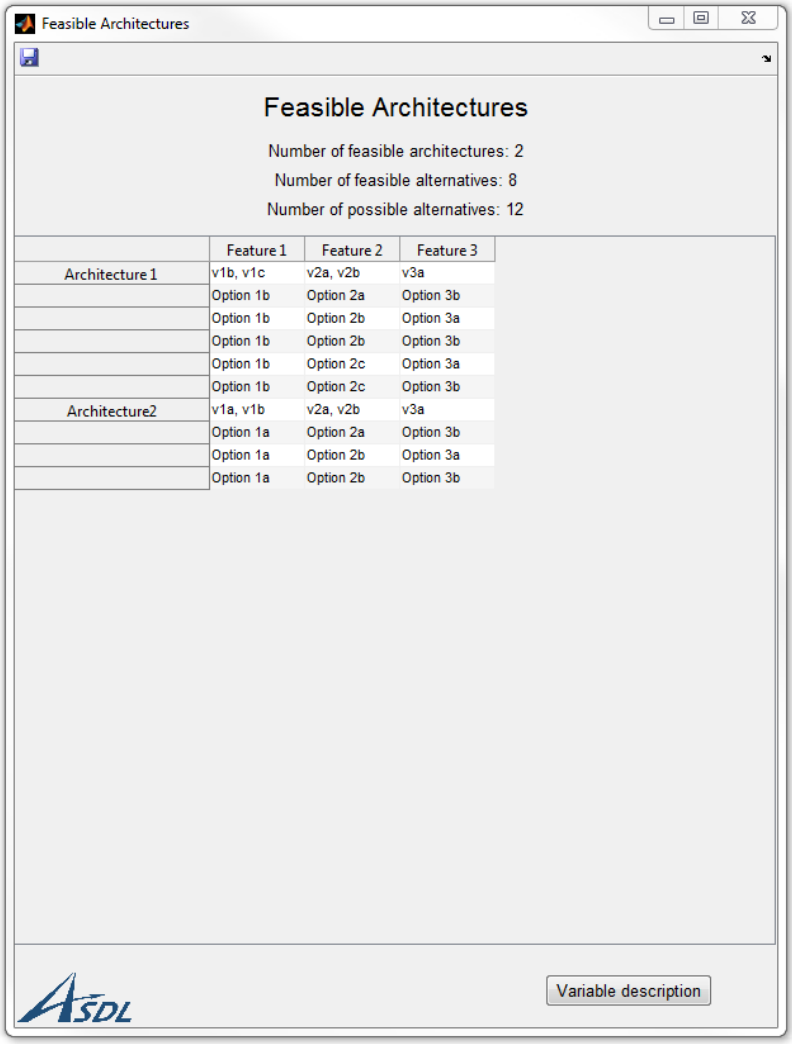



**Figure 172:** List of feasible architectures of ENVISAGE

# APPENDIX B

# DEVELOPMENT OF THE DESIGN FRAMEWORK FOR SUBORBITAL VEHICLES

## *B.1 Description of the Atmospheric Model*

This section describes the models used to determine each atmospheric parameter.

### B.1.1 Temperature

The model used for the temperature evolution $T$ is taken from the ISA [73]. The latter describes the temperature's behavior in the different atmospheric layers up to the mesopause. ISA provides a linear temperature model distribution through the atmospheric layers, as described in Equation 177, where the different coefficients are detailed in Table 60 for altitudes up to 90 km [9].

$$T(h) = T_1 + a\,(h - h_1) \tag{177}$$

**Table 60:** Coefficients of the atmospheric temperature model [9]

| **Layers** | **Altitude (km)** | | **Temperature (°C)** | | **Slope (°C/km)** |
|---|---|---|---|---|---|
| | $h_1$ | $h_{upper}$ | $T_1$ | $T_{upper}$ | $a$ |
| **Troposphere** | 0 | 11 | 15.0 | -56.5 | -6.5 |
| **Tropopause** | 11 | 20 | -56.5 | -56.5 | 0.0 |
| **Stratosphere** | 20 | 32 | -56.5 | -56.5 | +1.0 |
| | 32 | 47 | -44.5 | -2.5 | +2.8 |
| **Stratopause** | 47 | 51 | -2.5 | -2.5 | 0.0 |
| **Mesosphere** | 51 | 71 | -2.5 | -58.5 | -2.8 |
| | 71 | 85 | -58.5 | -86.2 | -2.0 |
| **Mesopause** | 85 | 90 | -86.2 | -86.2 | 0.0 |

In particular, this model provides standardized values such as the standard temperature and air pressure values at sea level: 15°C and 1,013.25 hPa. It is important to understand

that the difficulty to reach such high altitudes makes the modeling of the thermosphere very challenging. Moreover, the solar activity has a huge impact on the thermosphere's parameters. Studies performed by Rees [357] are used, where values given in standard conditions follow a linear evolution starting from 93 km. These equations have been embedded in order to create a consistent and complete temperature model, as displayed in Figure 173.

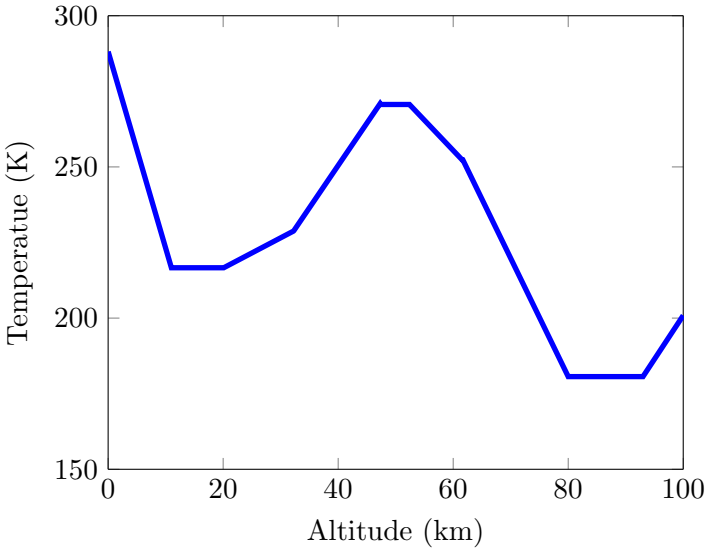



**Figure 173:** Temperature evolution with respect to altitude

### B.1.2 Pressure

The international barometric formula presented in Equation 178 provides the troposphere pressure $p$ (in hPa) in standard conditions, as suggested by the International Civil Aviation Organization (ICAO).

$$p(h) = 1013.25 \left(1 - \frac{0.0065h}{288.15}\right)^{5.255} \tag{178}$$

For the other layers, this equation is only partially valid. Hence, values and tendency curves are taken from dedicated studies, which approximate the pressure evolution with an exponential model [75, 357]. Figure 174 displays the pressure evolution with respect to altitude.

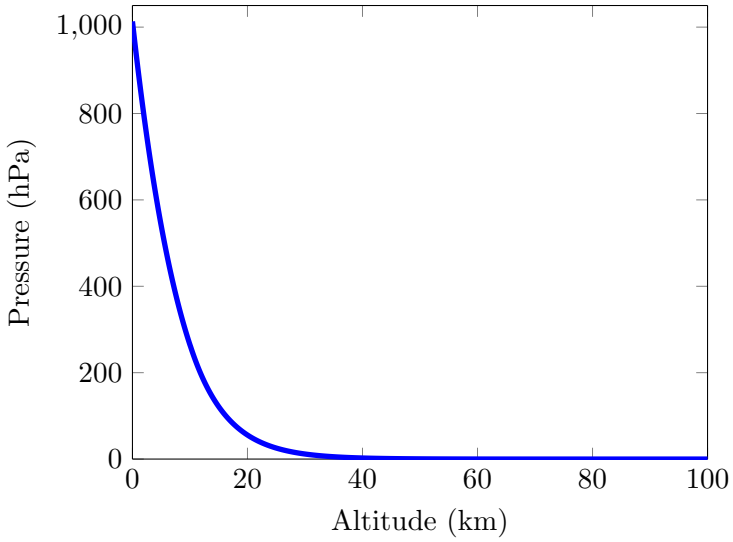



**Figure 174:** Pressure evolution with respect to altitude

### B.1.3 Density

The ideal gas law is used to calculate the density $\rho$, as presented in Equation 179, where $R$=8.314 J/K.mol is the universal gas constant and $M$=28.9644 kg/kmol, the air molar mass. Figure 175 displays the density evolution with respect to altitude.

$$\rho = \frac{Mp}{RT} = \frac{p}{287.058T} \tag{179}$$

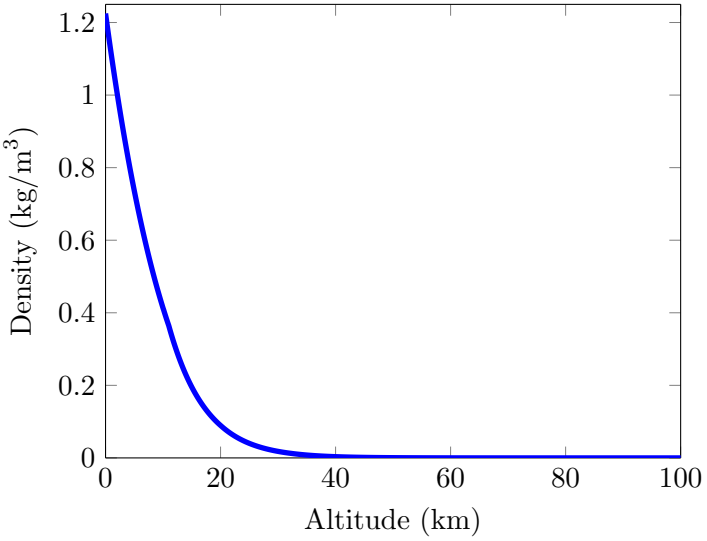



**Figure 175:** Density evolution with respect to altitude

### B.1.4 Acceleration of Gravity

The acceleration of gravity $g$ follows an empirical law presented in Equation 180, where $g_0 = 9.81\text{m.s}^{-2}$. For low altitude flights, the acceleration of gravity is usually considered as a constant and equal to $g_0$. However, since suborbital vehicles reach altitudes up to 100 km, its variations must be taken into account. Figure 176 illustrates how the acceleration of gravity changes with respect to altitude.

$$g(h) = \frac{g_0}{1 + \frac{2h}{6378000}} \tag{180}$$

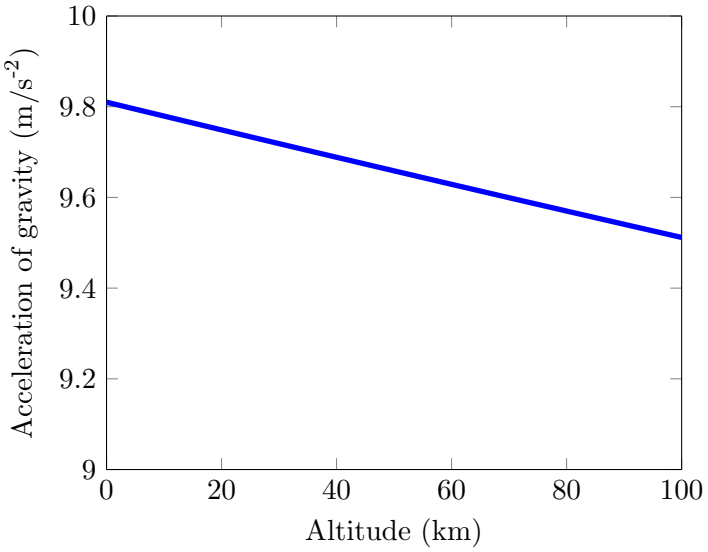



**Figure 176:** Acceleration of gravity evolution with respect to altitude

### B.1.5 Speed of Sound

The speed of sound $a$ is an important parameter since it relates velocity and Mach number. The speed of sound is only a function of temperature. Hence, using the aforementioned, temperature model, the speed of sound can be computed using Equation 181, where $R$ is the universal gas constant and $\gamma = 1.4$, the heat capacity ratio.

$$a = \sqrt{\gamma RT} \tag{181}$$

The speed of sound has the same behavior as the temperature with respect to altitude.

### B.1.6 Dynamic Viscosity

Sutherland's law is used to model the air dynamic viscosity $\mu$. The latter is completely independent of the pressure and only depends on the temperature. The viscosity model provided by Sutherland's law is presented in Equation 182, where $T_0 = 273.15$ K is the reference temperature, $S = 110.56$ K the Sutherland constant, and $\mu_0 = 1.71 \times 10^{-5}$ kg/(m.s) the reference viscosity [44].

$$\mu = \mu_0 \left(\frac{T}{T_0}\right)^{\frac{3}{2}} \left(\frac{T_0 + S}{T + S}\right) \tag{182}$$

Figure 177 illustrates how the viscosity changes with altitude.

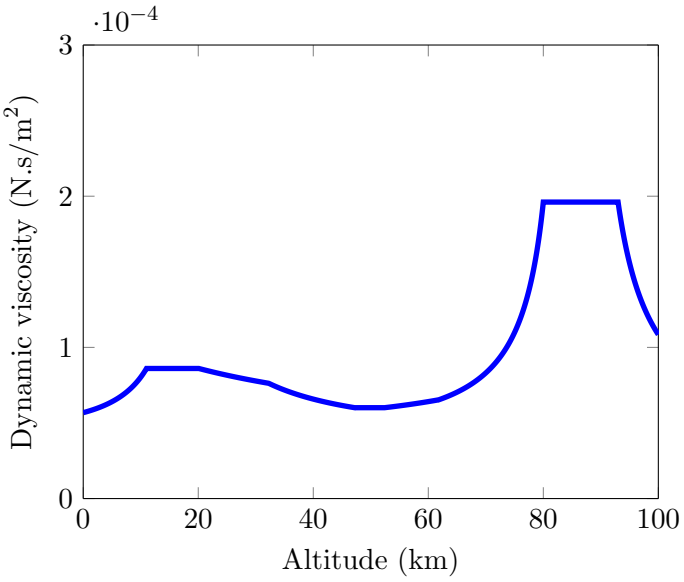



**Figure 177:** Dynamic viscosity evolution with respect to altitude

The presented atmospheric model has been implemented into a Matlab function presented in Appendix D.2.7.

## *B.2 Weight Estimation Module*

### B.2.1 Fuselage, Nose, and Thrust Structure

Brothers [55] developed an empirical model for body weight estimation based on aircraft, the Space Shuttle, and various expendable vehicles. This model decomposes the body into three parts: fuselage $W_{fus}$, thrust structure $W_{thrust-str}$, and nose $W_{nose}$. The fuselage

weight estimation model includes the weight of the fuselage with its base, as presented in Equation 183, where $S_{fus}$ is the fuselage area.

$$W_{fus} = 2.167 S_{fus}^{1.075} \tag{183}$$

Assuming an ellipsoidal nose, its weight can be estimated using Equation 184, where $S_{nose}$ is the surface area of the nosecone, $q_{max}$ the maximum dynamic pressure during the flight, and $d_{nose}$ the diameter of the nosecone base.

$$W_{nose} = S_{nose}\left(2.499 \times 10^{-4} q_{max} + 1.7008 + \left(3.695 q_{max} 10^{-5} - 3.252 \times 10^{-3}\right) d_{nose}\right) \tag{184}$$

Finally, the thrust structure weight model is presented in Equation 185, where $K_{thrust} = 1.949 \times 10^{-3}$ is the thrust structure constant, and $T_r$ the maximum vacuum thrust of the propulsion system.

$$W_{thrust-str} = K_{thrust} T_r^{1.0687} \tag{185}$$

### B.2.2 Thermal Protection System (TPS)

Brady et al. [47] propose an empirical model for the weight estimation of the external insulation $W_{TPS}$ of a lifting body second stage. This model is presented in Equation 186, where $S_{body}$ is the body planform area.

$$W_{TPS} = 1.51 S_{body} \tag{186}$$

### B.2.3 Wing

MacConochie and Klich [266] propose an empirical model for the wing weight $W_{wing}$ estimation derived from aircraft data and the Space Shuttle. The model is presented in Equation 187, where $n_u$ is the ultimate load factor, $W_{land}$ the landing mass, $S_{body}$ the body planform area, $S_{exp}$ the wing exposed area, $t_c$ the wing thickness-to-chord ratio, $b$ the wing span, $d_f$ the fuselage diameter, $\eta$ the wing/body efficiency factor, $K_w$ the exposed wing material/configuration constant, and $K_{ct}$ the wing carry-thru constant. According to MacConochie, $\eta = 0.2$, $K_w = 0.214$, and $K_{ct} = 0.05$.

$$W_{wing} = \left(\frac{n_u W_{land}}{1 + \frac{\eta S_{body}}{S_{exp}}}\right)^{0.386} \left(\frac{S_{exp}}{t_c}\right)^{0.572} \left(K_w b^{0.572} + K_{ct} d_f^{0.572}\right) \tag{187}$$

### B.2.4 Landing Gear

The landing gear weight $W_{lg}$ can be calculated using the models from MacConochie [266] and Brothers [55]. The complete model including nose and main gears, gear bays, and attachment is presented in Equation 188, where $W_{land}$ is the landing weight.

$$W_{lg} = 0.010784 W_{land}^{1.0861} + 0.0028 W_{land} \tag{188}$$

### B.2.5 Horizontal and Vertical Tails

Both the horizontal and the vertical tail weights $W_t$ are assumed to follow the same estimation method, which is a function of the tail area $S_t$ and the tail constant $K_t = 1.108$, as provided by MacConochie and Klich [266]. This model is presented in Equation 189.

$$W_t = K_t S_t^{1.24} \tag{189}$$

### B.2.6 Hydraulic System

The weight of the hydraulic system $W_{hyd}$ depends on size of the control surfaces $S_{cs}$ and the amount of the gimbaled thrust $T_r$ [266]. The model is presented in Equation 190.

$$W_{hyd} = 2.1 S_{cs} + 1.68 \times 10^{-4} T_r \tag{190}$$

### B.2.7 Parachute and Retrorockets

The weight of the parachute system is hard to estimate so historical data from both the Orion Multi-Purpose Crew Vehicle and the Soyuz are used as a reference. Indeed, for both spacecraft, the parachute represents around 4% of the vehicle's empty mass [85, 192]. The retrorockets that must be added to the spacecraft to enable a safe landing are also assumed to weigh 4% of the vehicle's empty mass [135]. In addition, a margin of 10% of the parachute and retrorocket weights is added to account for the pyrotechnic system.

### B.2.8 Reaction Control System (RCS)

The weight of the RCS, $W_{rcs}$, is described by MacConochie [266] as a function of the vehicle mass during the re-entry $W_{re}$ and length $l_f$, as presented in Equation 191. $K_{rcs} =$

$1.36 \times 10^{-4}$ is the RCS constant.

$$W_{rcs} = K_{rcs} W_{re} l_f \tag{191}$$

### B.2.9 Avionics

The weight of the avionic system $W_{av}$ is assumed to be proportional to the vehicle empty weight $W_e$, as suggested by Raymer [354]. In addition, the avionic technology is assumed to be similar to the one used for fighter aircraft so that Equation 192 can be used to predict its weight.

$$W_{av} = 0.055 W_e \tag{192}$$

### B.2.10 Environmental Control and Life Support System (ECLSS)

The weight of the ECLSS is provided by MacConochie as a function of the volume of the crew module $V_{crew}$, the number of days spent in space $N_{days}$, the weight of the avionic system $W_{av}$, as well as the number of passengers $n_{PAX}$ and the number of pilots $n_{pilots}$, as presented in Equation 193 [266].

$$W_{eclss} = 5.85 V_{crew}^{0.75} + 10.9 \left(n_{PAX} + n_{pilots}\right) N_{days} + 0.44 W_{av} \tag{193}$$

### B.2.11 Primary Power and Electrical Systems

The weights of the primary power system $W_{pp}$ and the electrical system $W_{elec}$ can be estimated based on the empirical model derived by MacConochie and Klich [266] using data from various aircraft and the Space Shuttle. The model is presented in Equation 194, where $K_{pc} = 0.712$ is the hydraulic system constant, $S_{cs}$ the area of the control surfaces, $K_{pe} = 9.7.10^{-5}$ the engine gimbal power constant, $T_r$ the vacuum thrust of the rocket engine, $K_{pb} = 0.405$ the battery power demand constant, and $W_{av}$ the avionics weight.

$$W_{pp} = K_{pc} S_{cs} + K_{pe} T_r + K_{pb} W_{av} \tag{194}$$

Similarly, the weight of the electrical system including electrical conversion and distribution is presented in Equation 195, where $K_{ecd} = 0.02$ represents the technology level of the electrical system.

$$W_{elec} = K_{ecd} W_{land} \tag{195}$$

### B.2.12 Flight Control

The flight control system includes actuators, surface control systems, and all systems useful to control the vehicle. Its weight $W_{fc}$ is provided by MacConochie and Klich [266], as presented in Equation 196, where $S_{cs}$ is the area of the control surfaces.

$$W_{fc} = 3.32 S_{cs} + 200 \tag{196}$$

### B.2.13 Seats and Accessories

The weight of the seats and other pilot/crew related items $W_{acc}$ is assumed to be 167 lbs per person, as suggested by MacConochie and Klich [266].

### B.2.14 Unused Propellant

Unused propellant includes multiple sources and can be estimated as a percentage of the total propellant mass, as described below:

- Start-up losses: 1% [425]
- In-flight losses and vents: 0.43% [266]
- Ascent reserve propellant: 0.75% [55]
- RCS reserve propellant: 0.75% [55]
- Residual propellant: 1.5% [47]

Combining all the losses, a safety margin for unused propellant is chosen as being 5% of the total propellant mass.

### B.2.15 Technology Reduction Factors

Since most of the aforementioned models have been developed between 1985 and 2000, there is a need to take into account recent enhancement in materials. For that purpose, Talay proposes a list of TRFs so that the updated weight $W_n$ of each component can be calculated using Equation 197, where $W_o$ is the original component weight [425]. The list of all TRFs is provided in Table 61.

$$W_n = W_o \, (1 - TRF) \tag{197}$$

**Table 61:** TRF for the various components [425]

| | Wing | Tail | Body | TPS | Landing gear | Avion-ics | ECLSS |
|---|---|---|---|---|---|---|---|
| **TRF** | 0.44 | 0.44 | 0.38 | 0.35 | 0.09 | 0.5 | 0.1 |

## *B.3 Aerodynamic Modeling*

This section aims at describing the development of the different models used in the aerodynamic module, which are mainly based on Roskma and Raymer's research [354, 369].

### B.3.1 Lift Coefficient Model

The vehicle lift coefficient $C_L$ can be decomposed into two different terms, as defined in Equation 198. In this equation, $\alpha$ represents the angle of attack and $C_{L_\alpha}$ the lift-curve slope.

$$C_L = C_{L_\alpha}\alpha \tag{198}$$

In this research, it is assumed that only the wing can produce lift. While the angle of attack depends on the flight conditions, the lift-curve slope can be modeled as a function of the Mach number and the wing configuration. In the subsonic regime, Raymer suggests the semi-empirical formula presented in Equation 199, where $AR$ is the aspect ratio, $\beta = \sqrt{1-M^2}$ the compressibility factor, $S_{exposed}$ the surface of the wing exposed to the air, $\Lambda$ the sweep angle, $\eta$ the airfoil efficiency factor, $F$ the fuselage lift factor [354], and $S$ the reference area.

$$C_{L_\alpha} = \frac{2\pi AR}{2+\sqrt{4+\frac{AR^2\beta^2}{\eta^2}\left(1+\frac{\tan^2\Lambda}{\beta^2}\right)}}\frac{S_{exposed}}{S}F \tag{199}$$

Raymer suggests a value of 0.95 for the airfoil efficiency factor and provides Equation 200 to calculate $F$, where $d$ is the fuselage diameter, and $b$ the wing span.

$$F = 1.07\left(1+\frac{d}{b}\right)^2 \tag{200}$$

This equation is assumed to be valid up to the drag divergence Mach number $M_{DD}$. While the latter has multiple definitions, the one proposed by Boeing is used [354]. Hence, $M_{DD}$

is defined as the Mach number at which the drag rise reaches 20 counts, which is usually about 0.08 Mach above the critical Mach number $M_c$. To compute $M_{DD}$, Raymer provides Figure 178, which links $M_{DD}$ to the sweep angle $\Lambda$ and the thickness-to-chord ratio $t_c$ [354].

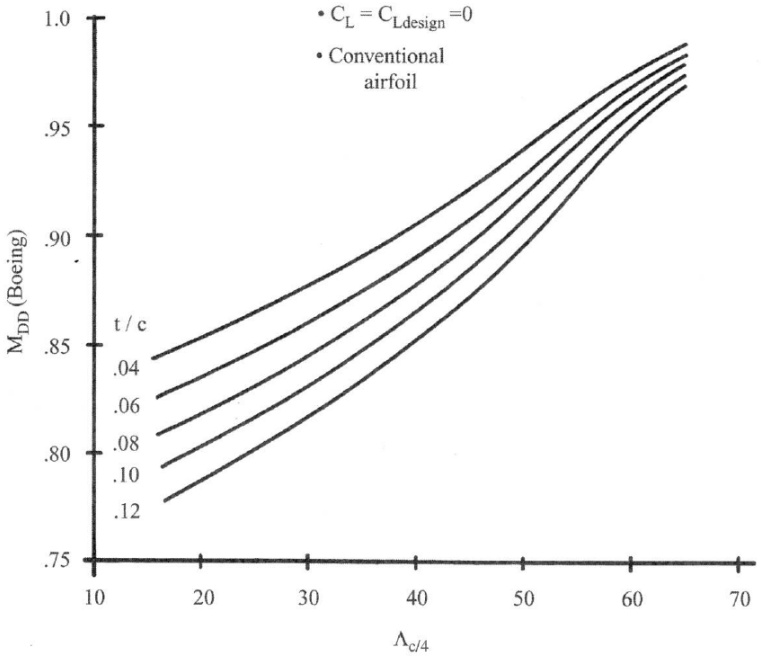



**Figure 178:** Wing drag-divergence Mach number [354]

Based on this figure, a surrogate model can be created in order to benefit from a parametric estimation. The result is provided in Equation 201, for which $R^2$ and $R^2_{adj}$ are greater than 0.998. Because the MRE has a mean of $10^{-3}$ and a standard deviation close to 0.25, the approximation was considered accurate enough for conceptual design level considerations.

$$M_{DD} = 0.79 + 4.10^{-3}\Lambda - 0.64t_c + 3.10^{-5}(\Lambda - 41.00)^2 + 0.01(\Lambda - 41.00)(t_c - 0.08) \quad (201)$$

For supersonic speed, usually beyond Mach 1.2, the theoretical lift-curve slope is defined in Equation 202, corrected by an efficiency factor $\eta_s$ to match actual data [354].

$$C_{L_\alpha} = \frac{4}{\sqrt{M^2 - 1}} \quad (202)$$

The efficiency factor $\eta_s$ is defined in Equation 203, where $C_{L_\alpha}(M_{DD})$ is the value of the lift-curve slope at the drag divergence Mach number.

$$\eta_s = 0.141 C_{L_\alpha}(M_{DD}) \tag{203}$$

According to Raymer, there are no good initial estimation methods for the transonic regime and it is suggested to smoothly link both the subsonic and the transonic regimes. To do so, a parametric interpolation is developed based on experimental data [354]. It is assumed that the maximum of the lift-curve slope occurs at Mach 0.98 and reaches a value 10% higher than the one at $M_{DD}$. In addition, the curve follows a second-order polynomial equation defined in Equation 204, where $a$, $b$, and $c$ are three constants to be determined.

$$C_{L_\alpha}(M) = aM^2 + bM + c \tag{204}$$

Based on this assumption, a matrix form of the problem is set, as presented in Equation 205.

$$\begin{bmatrix} C_{L_\alpha}(M_{DD}) \\ C_{L_\alpha}(1.2) \\ 1.1 C_{L_\alpha}(M_{DD}) \end{bmatrix} = \begin{bmatrix} M_{DD}^2 & M_{DD} & 1 \\ 1.2^2 & 1.2 & 1 \\ 0.98^2 & 0.98 & 1 \end{bmatrix} \begin{bmatrix} a \\ b \\ c \end{bmatrix} \tag{205}$$

Upon solving this system, the three constants $a$, $b$, and $c$ are determined and presented in Equations 206 to 208.

$$a = -22.3 \frac{55 C_{L_\alpha}(M_{DD}) + 50 M_{DD} C_{L_\alpha}(1.2) - 49 C_{L_\alpha}(1.2) - 55 M_{DD} C_{L_\alpha}(M_{DD})}{250 M_{DD}^2 + 294 - 545 M_{DD}} \tag{206}$$

$$b = 0.45 \frac{2761 C_{L_\alpha}(M_{DD}) - 2401 C_{L_\alpha}(1.2) - 2750 C_{L_\alpha}(M_{DD}) Mdd^2 + 2500 M_{DD}^2 C_{L_\alpha}(1.2)}{250 M_{DD}^2 + 294. - 545 M_{DD}} \tag{207}$$

$$c = -0.09 \frac{-3234 C_{L_\alpha}(M_{DD}) - 12005 M_{DD} C_{L_\alpha}(1.2) + 19800 M_{DD} C_{L_\alpha}(M_{DD})}{250 M_{DD}^2 + 294. - 545 M_{DD}} \times \left(-16500 C_{L_\alpha}(M_{DD}) M_{DD}^2 + 12250 M_{DD}^2 C_{L_\alpha}(1.2)\right) \tag{208}$$

Figure 179 shows the results of the previous model by plotting the lift-curve slope as a function of the Mach number and the sweep angle. The plot confirms that, the higher the sweep angle, the lower the lift coefficient at a given angle of attack.

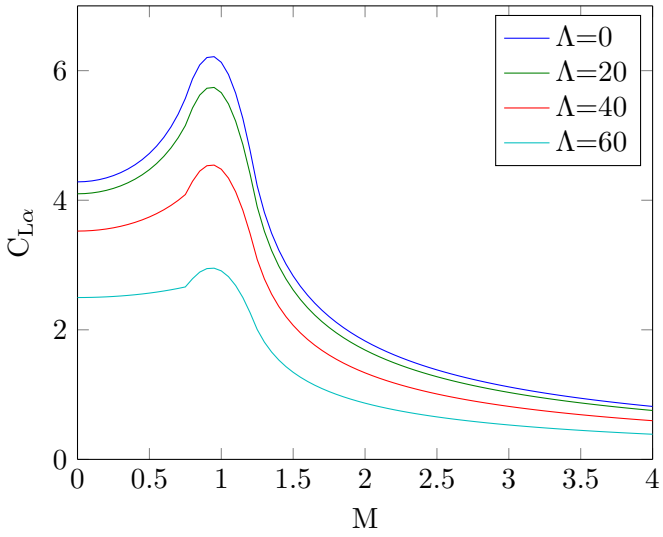



**Figure 179:** $C_{L_\alpha}$ with respect to the Mach number and the sweep angle

### B.3.2 Subsonic and Supersonic Drag Coefficient Models

Roskam decomposes the drag coefficient into 11 components: wing drag coefficient ($C_{D_{wing}}$), fuselage drag coefficient ($C_{D_{fus}}$), empennage drag coefficient ($C_{D_{emp}}$), nacelle/pylon drag coefficient ($C_{D_{np}}$), flap drag coefficient ($C_{D_{flap}}$), landing gear drag coefficient ($C_{D_{gear}}$), canopy/windshield drag coefficient ($C_{D_{cw}}$), store drag coefficient ($C_{D_{store}}$), trim drag coefficient ($C_{D_{trim}}$), interference drag coefficient ($C_{D_{int}}$), and miscellaneous drag coefficient ($C_{D_{misc}}$). The latter includes the drag caused by speed brakes, struts, inlets, antennas, gaps, and surface roughness. In addition to this first decomposition, the model for the drag coefficient is specific to each flight regime: subsonic and supersonic.

#### *B.3.2.1 Wing Drag Coefficient*

The wing drag coefficient is decomposed into two components, as shown in Equation 209. In this equation, $C_{D_{0,w}}$ is the wing zero-lift drag coefficient and $C_{D_{L,w}}$ is the wing drag coefficient due to lift.

$$C_{D_{wing}} = C_{D_{0,w}} + C_{D_{L,w}} \tag{209}$$

**Subsonic wing zero-lift coefficient**

For subsonic speeds, $C_{D_{0,w}}$ is modeled in Equation 210, where $R_{wf}$ is the wing/fuselage interference factor, $R_{LS}$ the lifting surface correction factor, $C_{f,w}$ the wing skin friction coefficient, $L'$ the airfoil thickness location parameter, $t_c$ the thickness ratio, $S_{wet,w}$ the wing wetted area, and $S$ the reference surface area. This paragraph aims at defining each of these parameters.

$$C_{D_{0,w}} = R_{wf}.R_{LS}.C_{f,w}.\left(1 + L't_c + 100t_c^4\right)\frac{S_{wet,w}}{S} \tag{210}$$

$R_{wf}$ is modeled as a function of the fuselage Reynolds number by Finck [148], as presented in Figure 180.

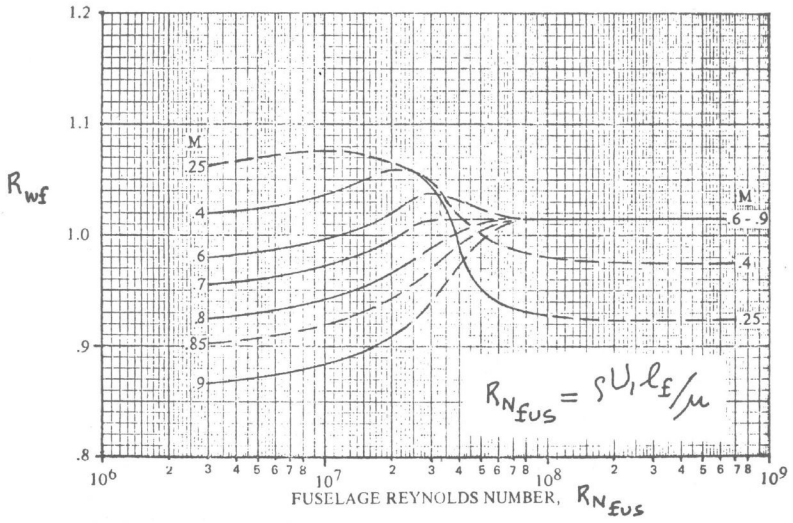



**Figure 180:** Wing-fuselage interference factor [369]

Based on this plot, a mathematical model is established that enables a more parametric estimation of $R_{wf}$. Around 275 data points have been selected from the seven curves in order to create a RSE. $R^2$ and $R^2_{adj}$ are around 0.95. In addition, the MRE shows a mean equal to 0.01 and a standard deviation close to 1.1. The values of these parameters confirm

the accuracy of the developed model presented in Equation 211.

$$\begin{aligned} R_{wf} = {} & 1.06 - 0.08M - 0.004\left(\log \mathrm{Re} - 17.42\right)^2 + 0.15\left(\log \mathrm{Re} - 17.41\right)\left(M - 0.59\right) \\ & - 0.35\left(M - 0.59\right)^2 + 0.001\left(\log \mathrm{Re} - 17.42\right)^3 - 0.001\left(\log \mathrm{Re} - 17.42\right)^3\left(M - 0.59\right) \\ & + 0.42\left(M - 0.59\right)^3\left(\log \mathrm{Re} - 17.42\right) \end{aligned} \tag{211}$$

The wing skin friction coefficient $C_{f,w}$ can be calculated by first determining its value in an incompressible flow and then by evaluating the compressible effect. Assuming that the flow is entirely turbulent, the incompressible skin-friction coefficient $C_{f,inc}$ can be modeled using Equation 212 [466].

$$C_{f,inc} = \frac{0.074}{\mathrm{Re}^{0.2}} \tag{212}$$

However, in addition to the Reynolds number, the Mach number and the temperature also have an effect on the skin friction coefficient. A method for determining this effect has been established by Sommer and Short [406]. The ratio between the actual skin friction coefficient $C_f$ and the incompressible skin friction coefficient $C_{f,inc}$ can be found using the free-stream temperature $T_\infty$ and the free-stream Mach number $M_\infty$. The system of equations to be solved is defined in Equation 213.

$$\begin{cases} \frac{T'}{T_\infty} = 1 + 0.1151 M_\infty^2 \\ \frac{R'}{R_\infty} = \left(\frac{T_\infty}{T'}\right)^{2.5} \frac{T'+216}{T_\infty+216} \\ \frac{C_f}{C_{f,inc}} = \left(\frac{T_\infty}{T'}\right)\left(\frac{R_\infty}{R'}\right)^{0.2} \end{cases} \tag{213}$$

Based on this model, the wing skin friction coefficient $C_{f,w}$ is modeled as a function of the wing's Reynolds number with its mean geometric chord.

The lifting surface correction factor $R_{LS}$ is defined in Figure 181 provided by Roskam [369]. Based on this plot, a mathematical model is established that enables a more parametric estimation of $R_{LS}$. Around 100 data points have been selected from the four curves in order to create a RSE. $R^2$ and $R^2_{adj}$ are equal to 0.9971 and 0.9970, respectively. In addition, the MRE shows a mean equal to 0.002 and a standard deviation close to 0.67, which confirms a relatively good accuracy.

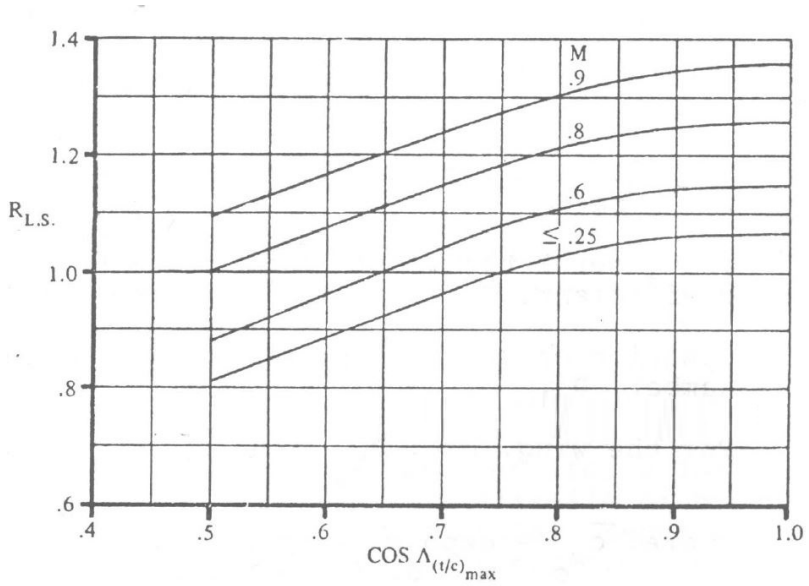



**Figure 181:** Lifting surface correction factor [369]

Hence, $R_{LS}$ can be modeled in Equation 214, where $\Lambda$ is the sweep angle at the location of the maximum thickness-to-chord ratio.

$$R_{LS} = 0.32 + 0.57\cos\Lambda + 0.52\max(M, 0.25) - 0.89(\cos\Lambda - 0.74)^2 + 0.72(\max(M, 0.25) - 0.64)^2 \quad (214)$$

For traditional supersonic airfoils such as the NACA 64A, the location of the maximum thickness-to-chord ratio ($t_c$) is behind the third of the chord [354]. Hence, according to Roskam, $L' = 1.2$ [369]. In addition, the wing wetted area $S_{wet,w}$ is assumed to be twice the wing reference area $S$.

### Subsonic wing drag coefficient due to lift

For subsonic speeds, $C_{D_{L,w}}$ is modeled using Equation 215. In this equation, the last two terms represent the drag due to the linear twist of the wing, which is neglected in this study.

$$C_{D_{L,w}} = \frac{C_{L,w}^2}{\pi ARe} + 2\pi C_{L,w}\epsilon_t v + 4\pi^2\epsilon_t^2 w \quad (215)$$

As suggested by Roskam, $C_{L,w}$ can be modeled at a conceptual design level using Equation 216 [369].

$$C_{L,w} = 1.05 C_L \tag{216}$$

The span efficiency factor $e$ has been modeled by equations from various sources. A review and comparison of these models are provided by Nita and Scholz [321]. In this review, Howe's model was found to be one of the most accurate and general formulations [209]. Assuming that no engines are mounted on the wings, the span efficiency factor is presented in Equation 217, where $AR$ is the wing aspect ratio, $\lambda$ the wing taper ratio, $\phi$ the wing sweep angle, and $t_c$ the wing thickness-to-chord ratio. In addition, it is assumed that there is no twist angle so that $\epsilon_t = 0$.

$$e = \left(1 + 0.12 M^6\right)^{-1} \left(1 + \frac{0.142 + 0.005\left(1 + 1.5\left(\lambda - 0.6\right)^2\right) AR \left(10 t_c\right)^{0.33}}{\cos^2 \phi} + \frac{0.1}{\left(4 + AR\right)^{0.8}}\right)^{-1} \tag{217}$$

**Supersonic wing zero-lift drag coefficient**

Assuming wings with a supersonic leading edge, the supersonic wing zero-lift drag coefficient $C_{D_{0,w}}$ is defined in Equation 218, where $C_{f_w}$ is the wing skin friction coefficient, $S_{wet}$ the wing wetted area, $C_{D_{LE}}$ the leading edge pressure drag coefficient, $\beta = \sqrt{|1 - M^2|}$ the compressibility factor, and $t_{c_{eff}}$ the effective thickness-to-chord ratio.

$$C_{D_{0,w}} = C_{f_w} \frac{S_{wet}}{S} + C_{D_{LE}} + \frac{16}{3\beta} t_{c_{eff}}^2 \tag{218}$$

The effective thickness-to-chord ratio $t_{c_{eff}}$ is defined in Equation 219, where $c_{bw}$ is the mean aerodynamic chord.

$$t_{c_{eff}} = \frac{\left(\int_0^{b/2} t_c^2 c_{bw} dy\right)^{1/2}}{\left(\frac{S}{2}\right)^2} \tag{219}$$

$C_{D_{LE}}$ is modeled by Roskam using Equation 220 [369]. In this equation, $r_{LE}$ is the

leading edge radius.

$$C_{D_{LE}} = 1.28 \frac{M^3 \cos^6 \Lambda}{1 + M^3 \cos^3 \Lambda} \frac{2 r_{LE} \frac{b}{cos\Lambda}}{S} \tag{220}$$

**Supersonic wing drag coefficient due to lift**

According to Roskam, the supersonic wing drag coefficient due to lift can be evaluated using Equation 221 [369].

$$C_{D_{L,w}} = \frac{C_{D_L}}{C_L^2} C_L^2 \tag{221}$$

To determine the ratio on the right hand side of Equation 221, the planform shape parameter $p = \frac{S}{bc_r}$ and the planform slenderness parameter $\frac{b}{2c_r}$ must first be calculated. From there, the ratio $\frac{C_{D_L}}{C_L^2}$ can be determined using Figure 182 provided by Roskam [369].

Since these two curves are very close, they have been merged. The resulting curve is defined as the mean of the two original curves in order to account for the uncertainty in the type of leading edges. The curve is then modeled by a piecewise linear equation, as presented in Equation 222.

$$\begin{cases} \pi A \frac{C_{D_L}}{C_L^2} \frac{p}{1+p} & = 0.55 \quad \text{if } \frac{\beta b_w}{2c_{r_w}} < 0.375 \\ \pi A \frac{C_{D_L}}{C_L^2} \frac{p}{1+p} & = 0.9333 \frac{\beta b_w}{2c_{r_w}} + 0.23 \quad \text{if } \frac{\beta b_w}{2c_{r_w}} > 0.375 \end{cases} \tag{222}$$

*B.3.2.2 Fuselage Drag Coefficient*

Similarly to the wing, the fuselage drag coefficient can be decomposed into two different components, as described in Equation 223. $C_{D_{0,fus}}$ is the fuselage zero-lift drag coefficient and $C_{D_{L,fus}}$ is the fuselage drag coefficient due to lift.

$$C_{D_{fus}} = C_{D_{0,fus}} + C_{D_{L,fus}} \tag{223}$$

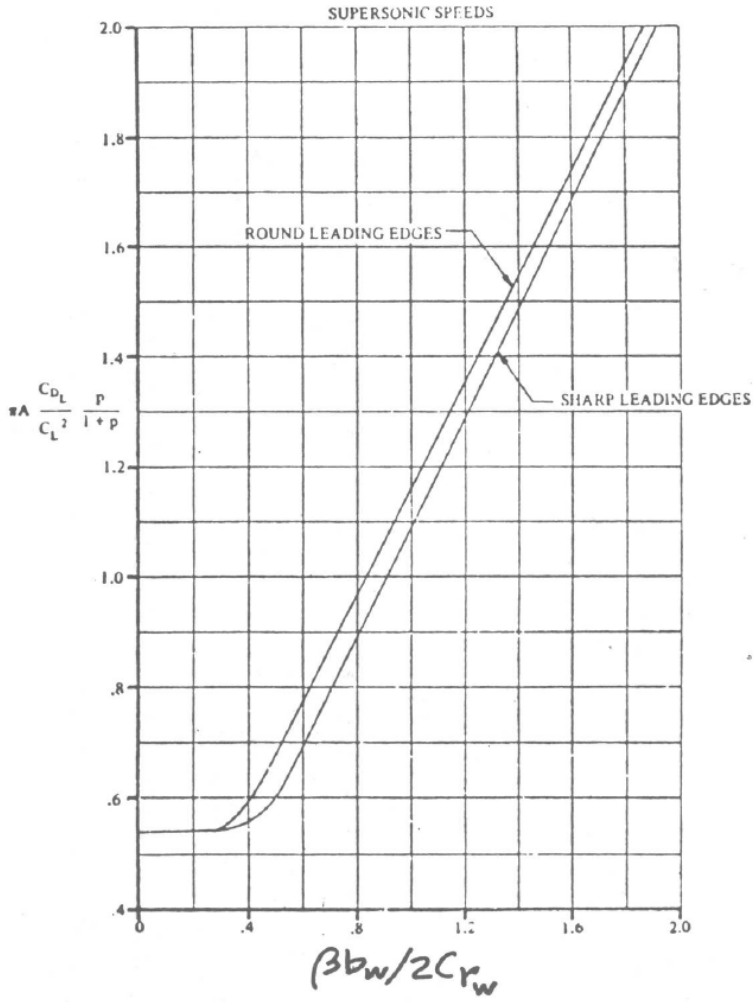



**Figure 182:** Supersonic drag due to lift [369]

**Subsonic fuselage zero-lift drag coefficient**

In the subsonic regime, Roskam models the fuselage zero-lift drag coefficient using Equation 224 [369].

$$C_{D_{0,fus}} = R_{wf} C_{f_{fus}} \left(1 + 60\left(\frac{l_f}{d_f}\right)^{-3} + 0.0025\left(\frac{l_f}{d_f}\right)\right)\frac{S_{wet,fus}}{S} + C_{D_{b,fus}} \tag{224}$$

For winged vehicles, $R_{wf}$ can be calculated using Equation 211. Otherwise, it can be set to 1. $C_{f_{fus}}$ can be calculated using Equations 212 and 213 with $l_f$ the reference length for the fuselage. $C_{D_{b,fus}}$ is the fuselage base drag coefficient as described in Equation 225 [369], where $S_{fus}$ is the fuselage maximum frontal area and $C_{D_{0,fus-base}}$ the zero-lift drag coefficient of the fuselage exclusive of the base. The latter corresponds to the first term on the right

hand side of Equation 224.

$$C_{D_{b,fus}} = 0.029 \left(\frac{d_b}{d_f}\right)^3 \left(C_{D_{0,fus-base}} \frac{S_{fus}}{S}\right)^{-0.5} \frac{S_{fus}}{S} \tag{225}$$

**Subsonic fuselage drag coefficient due to lift**

The subsonic fuselage drag coefficient due to lift $C_{D_{L,fus}}$ is presented in Equation 226 [369]. $\alpha$ is the angle of attack and $S_{bfus}$ the fuselage base area.

$$C_{D_{L,fus}} = \frac{2\alpha^2 S_{bfus}}{S} + \eta c_{d_c} \alpha^3 \frac{S_{plf_{fus}}}{S} \tag{226}$$

The ratio of the drag of a finite cylinder to the drag of an infinite cylinder $\eta$ is modeled by Roskam as a function of the body fineness ratio $\frac{l_f}{d_f}$ through Figure 183 [369].

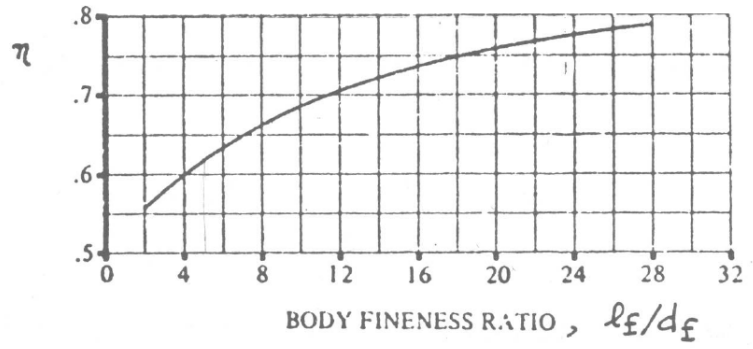



**Figure 183:** Ratio of the drag of a finite cylinder to the drag an infinite cylinder [369]

The curve in Figure 183 is mathematically modeled using a general power model as described in Equation 227. This model has a $R^2$ and a $R^2_{adj}$ above 0.995. The MRE also provides a mean of -0.0003 and a standard deviation of 0.7.

$$\eta = 0.4964 \left(\frac{l_f}{d_f}\right)^{0.1407} \tag{227}$$

Similarly, $c_{d_c}$ is presented in Figure 184 [369]. The curve in Figure 184 is mathematically modeled using two different equations based on Gaussian models, as presented in Equation 228. This model has a $R^2$ and a $R^2_{adj}$ above 0.9994. The MRE also provides a mean around 0.5 and a standard deviation around 0.35.

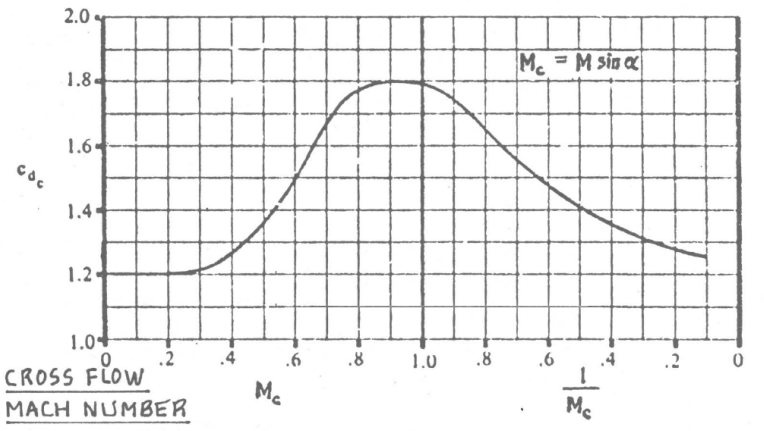



**Figure 184:** Steady state cross-flow drag coefficient for 2D circular cylinders [369]

In these equations, $M_c = M \sin\alpha$ represents the cross flow Mach number.

$$c_{d_c}(M_c < 1) = 0.1986 \exp\left(-\left(\frac{M_c - 1.05}{0.121}\right)^2\right) + 0.567 \exp\left(-\left(\frac{M_c - 0.8343}{0.2867}\right)^2\right) + 7.182 \exp\left(-\left(\frac{M_c - 218.3}{163.2}\right)^2\right)$$

$$c_{d_c}(M_c > 1) = 0.2642 \exp\left(-\left(\frac{M_c^{-1} - 1.062}{0.3659}\right)^2\right) + 3.87 \exp\left(-\left(\frac{M_c^{-1} - 9.531}{8.874}\right)^2\right) \tag{228}$$

**Supersonic fuselage zero-lift drag coefficient**

The supersonic fuselage zero-lift drag coefficient is modeled using Equation 229 from Roskam [369], where $C_{f_{fus}}$ is the turbulent flat plate skin-friction coefficient of the fuselage, $C_{D_{A(NC)}}$ the interference drag coefficient acting on the aft-fuselage due to the center fuselage, $C_{D_{b,fus}}$ the fuselage base drag coefficient, and $C_{D_{N2}}$ and $C_{D_A}$ the wave drag coefficients of the fuselage nose and afterbody, respectively. They can be determined using Figure 185, assuming both the nose and the afterbody are conical.

$$C_{D_{0,fus}} = C_{f_{fus}} \frac{S_{wet,fus}}{S} + \left(C_{D_{N2}} + C_{D_A} + C_{D_{A(NC)}} + C_{D_{b,fus}}\right) \frac{S_{fus}}{S} \tag{229}$$

To enable a rapid parametric analysis, data from Figure 185 are modeled through Equation 230. The latter is a surrogate model generated with a $R^2$=0.99.

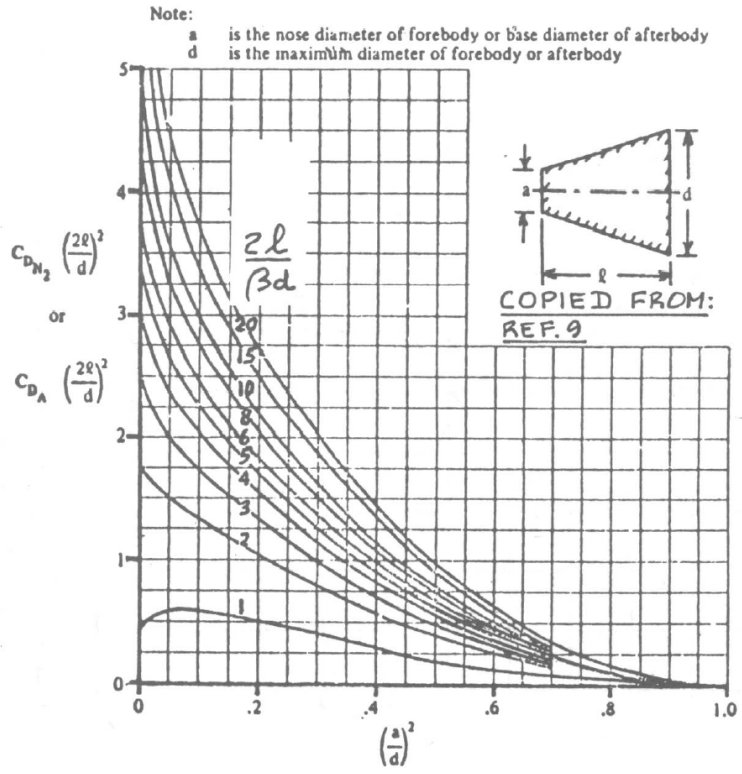



**Figure 185:** Drag of slender bodies, conical forebodies, and conical afterbodies [369]

$$C_{D_{N2/A}}\left(\frac{2l}{d}\right)^2 = 2.28 + 0.068\frac{2l}{\beta d} - 4.298\left(\frac{a}{d}\right)^2 - 4.10^{-3}\left(\frac{2l}{\beta d} - 9.23\right)^2 - 0.27\left(\frac{2l}{d} - 9.23\right)\left(\left(\frac{a}{d}\right)^2 - 0.39\right) + 6.81\left(\left(\frac{a}{d}\right)^2 - 0.39\right)^2 + 0.01\left(\frac{2l}{d} - 9.23\right)^2\left(\left(\frac{a}{d}\right)^2 - 0.39\right) + 0.29\left(\frac{2l}{d} - 9.23\right)\left(\left(\frac{a}{d}\right)^2 - 0.39\right)^2 - 5.78\left(\left(\frac{a}{d}\right)^2 - 0.39\right)^3 \quad (230)$$

$C_{D_{A(NC)}}$ is the interference drag coefficient acting on the aft-fuselage due to the center-fuselage. It can be found in Figure 186, which has been modeled using Equation 231 with a $R^2$ of 0.998.

$$C_{D_{A(NC)}}\left(\frac{2l_A}{d}\right)^2 = \exp\left[0.69 - 0.69\frac{l_A}{d} - 1.49\frac{l_C}{l_A} + 0.28\left(\frac{l_N}{l_A} - 1.15\right)^2 + 0.33\left(\frac{l_N}{l_A} - 1.15\right)\left(\frac{l_C}{l_A} - 0.60\right) + 0.61\left(\frac{l_C}{l_A} - 0.60\right)^2\right] \quad (231)$$

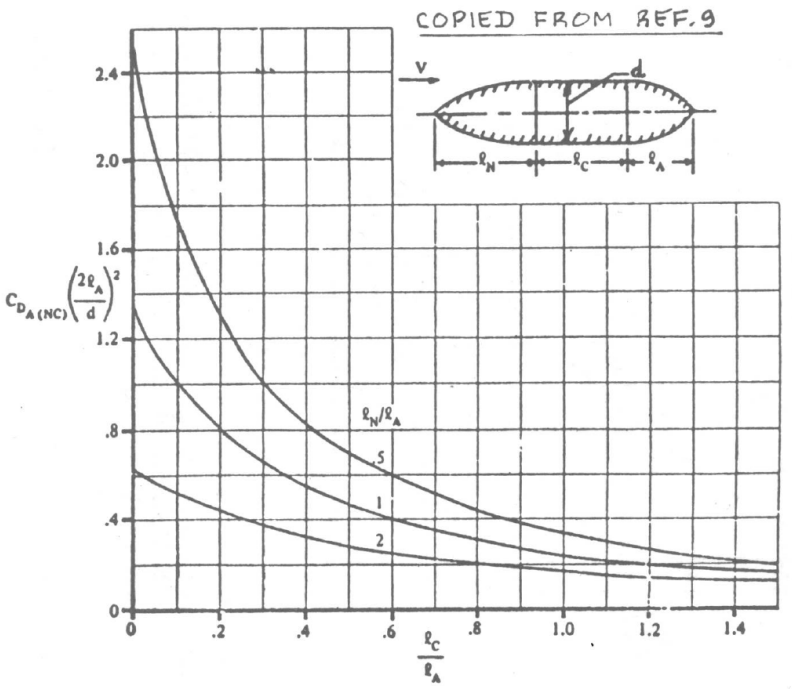



**Figure 186:** Fuselage interference drag [369]

$C_{D_{b,fus}}$ is modeled in Figure 187 and has been approximated using the double exponential model, as detailed in Equation 232. The $R^2$ of this surrogate model is 0.999.

$$C_{D_{b,fus}} = 0.2508 \exp\left(-0.5768M\right) + 0.104 \exp\left(-0.2178M\right) \tag{232}$$

**Supersonic fuselage drag coefficient due to lift**

The supersonic fuselage drag coefficient due to lift can be calculated using Equation 233 given by Roskam [369], where $S_{b,fus}$ is the fuselage base area, $c_{d_c}$ the steady state cross-flow drag coefficient for two-dimensional circular cylinders, and $S_{plf,fus}$ the fuselage planform area.

$$C_{D_{L,fus}} = 2\alpha^2 \frac{S_{b,fus}}{S} + c_{d_c} \frac{S_{plf,fus}}{S} \alpha^3 \tag{233}$$

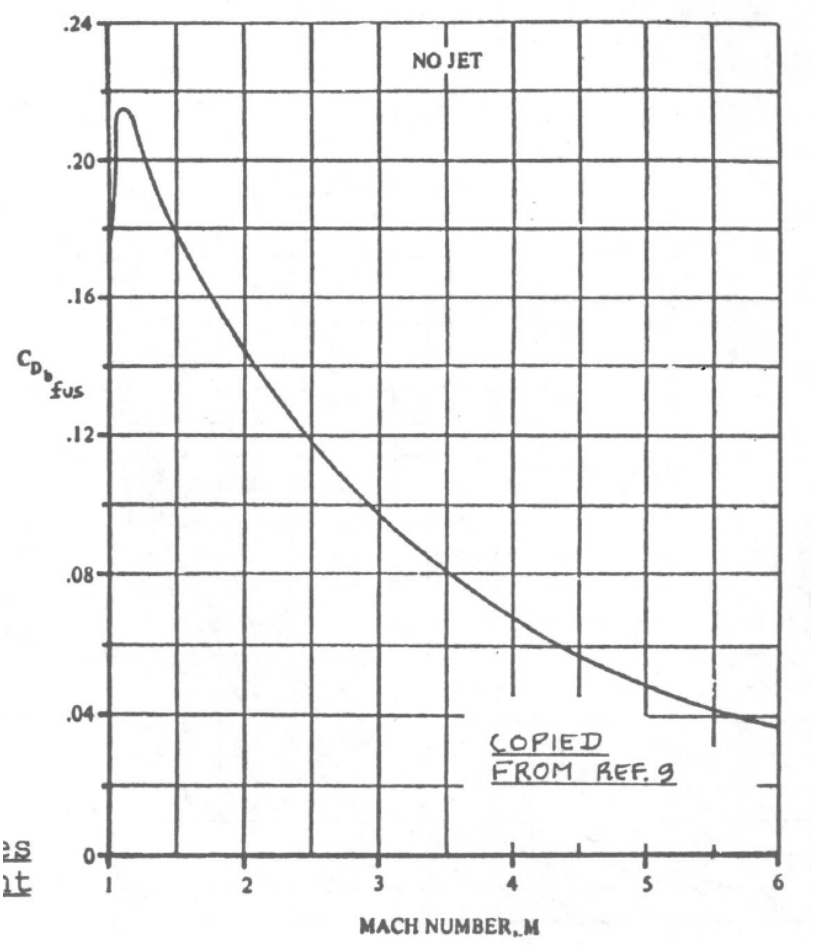



**Figure 187:** Base drag coefficient for bodies of revolution [369]

*B.3.2.3 Empennage Drag Coefficient*

Similar to the wing and the fuselage, the empennage drag coefficient can be decomposed into two different components, as described in Equation 234. $C_{D_{0,emp}}$ is the empennage zero-lift drag coefficient and $C_{D_{L,emp}}$ is the empennage drag coefficient due to lift.

$$C_{D_{emp}} = C_{D_{0,emp}} + C_{D_{L,emp}} \tag{234}$$

However, all horizontal surfaces are not designed to create lift and consequently their induced drag component is assumed to be negligible. In addition, vertical surfaces are assumed to be symmetric and do not create lift either. Hence, their induced drag component is also negligible.

**Subsonic empennage zero-lift drag coefficient**

The calculation of the zero-lift drag coefficient of horizontal tail surfaces, canard, and vertical tail surfaces is similar and can be done using Equation 210. In this equation, $S_{wet,w}$ is replaced by $S_{wet,emp}$, $C_{f,w}$ is replaced by $C_{f,emp}$, $R_{wf}$ is set to 1, and $t_c$ is the maximum thickness-to-chord ratio of the empennage. Finally, $R_{LS}$ is determined using Equation 214.

**Supersonic empennage zero-lift drag coefficient**

The supersonic drag coefficient of all empennage surfaces can be calculated using Equation 235.

$$C_{D_{0,emp}} = C_{D_{emp_f}} + C_{D_{emp,wave}} \tag{235}$$

The first term on the right hand side of Equation 235 can be found using Equation 236, where $C_{f_{emp}}$ can be calculated using Equation 213 and substituting wing parameters for empennage parameters.

$$C_{D_{emp_f}} = C_{f_{emp}} \frac{S_{wet,emp}}{S} \tag{236}$$

Moreover, $C_{D_{emp,wave}}$ can be determined following the same procedure as for the wing by replacing wing parameters by empennage parameters.

#### *B.3.2.4 Nacelle/Pylon Drag Coefficient*

Raymer provides an approach to evaluate the nacelle drag coefficient based on its component form factor $FF_n$, as presented in Equation 237, where $Q_n$ is the interference factor between the nacelle and the fuselage, $S_{wet,n}$ the wetted area of the nacelle, and $C_{f,n}$ the nacelle skin-friction coefficient.

$$C_{D_{np}} = C_{f,n} FF_n Q_n \frac{S_{wet,n}}{S} \tag{237}$$

According to Raymer, $FF_n$ can be determined using Equation 238, where $d$ is the nacelle diameter and $l$ its length [354]. In addition, as suggested by Raymer, $Q_n$ can be set to a value equal to 1.3.

$$FF_n = 1 + \frac{0.35d}{l} \tag{238}$$

**Windmilling drag and propeller drag coefficients**

If the engine is not running, a drag increment due to engine windmilling must be considered. It is defined in Equation 239 provided by Roskam [369]. In this equation, $S_{noz}$ is the nozzle section area and $\lambda_{noz}$ is the ratio of the average flow velocity in the nozzle to the steady state flight speed. Roskam provides its value as a function of the inlet diameter $d_{inl}$ and the type of jet engine installed: 0.25 for turbojet engines and 0.42 for low bypass turbofan engines.

$$\Delta C_{D_{wmj}} = 0.0785 \frac{d_{inl}^2}{S} + \left( \frac{2}{1 + 0.16M^2} \right) \lambda_{noz} \left(1 - \lambda_{noz}\right) \frac{S_{noz}}{S} \tag{239}$$

*B.3.2.5 Flap Drag Coefficient*

It is assumed that flaps are only used for take-off, approach, and landing. Two different deflection angles $\delta_f$ are considered: 15° for take-off and approach and 30° for landing. McCormick [287] then provides Equation 240 to estimate the drag increment due to the flaps, where $S_{flap}$ and $c_{flap}$ are the surface and the length of the flap in the longitudinal direction, respectively, and $c$ the wing chord.

$$\Delta C_{D_{flap}} = 0.9 \left( \frac{c_{flap}}{c} \right)^{1.38} \left( \frac{S_{flap}}{S} \right) \sin^2 \delta \tag{240}$$

*B.3.2.6 Landing Gear Drag Coefficient*

If a landing gear is needed, it is assumed to be tricycle with one nose landing gear and two main landing gears. In this case, the landing gear drag coefficient can be decomposed into two different terms, as presented by Roskam in Equation 241 [369]. The subscript $m$ represents the main landing gear, and the subscript $n$ represents the nose landing gear.

$$C_{D_{gear}} = \left( C_{D_{gear_{C_L=0,n}}} + p_n C_L \right) \frac{S_{gear,n}}{S} + \left( C_{D_{gear_{C_L=0,m}}} + p_m C_L \right) \frac{S_{gear,m}}{S} \tag{241}$$

$C_{D_{gear_{C_L=0}}}$ is defined as the zero-lift drag coefficient of the landing gear based on the reference area $S_{gear}$ defined by the product of the tire diameter $D_t$ and the tire width $b_t$. Based on the data provided by Roskam, a mean value is taken for the two zero-lift drag

coefficients, as shown in Equation 242.

$$\begin{cases} C_{D_{gear_{C_L=0,n}}} = 0.5 \\ C_{D_{gear_{C_L=0,m}}} = 1.3 \end{cases} \tag{242}$$

In addition, Roskam also provides models for $p_m$ and $p_n$ as described in Equation 243.

$$\begin{cases} p_n = -0.25 C_{D_{gear_{C_L=0,n}}} \\ p_m = -0.4 C_{D_{gear_{C_L=0,m}}} \end{cases} \tag{243}$$

Finally, in order to determine $S_{gear}$, a model is needed to compute the tire dimensions. Raymer provides a way to estimate the diameter and the width of aircraft tires as a function of the weight carried by the wheel [354]. At a conceptual design level, it is assumed that the main landing gear carries 90% of the aircraft weight [427]. Based on this assumption, Equation 244 can be used to estimate the size of the tires. Table 62 shows the values of the different parameters. Finally, it is assumed that there are two tires per landing gear. $W_{to}$ corresponds to the Take-Off Gross Weight (TOGW) for Horizontal Take-Off (HTO) vehicles and to the landing weight for Vertical Take-Off (VTO) vehicles. Then, according to Raymer, the nose landing gear tires measure around 60% of the main landing gear tires.

$$L_t / b_t = A W_{to}^B \tag{244}$$

**Table 62:** Parameters for dimension estimation of the main landing gear tires [354]

| Landing gear | $W_{to}$ | $D_t$ (cm) | | $b_t$(cm) | |
|---|---|---|---|---|---|
| | | A | B | A | B |
| Main | $0.45W$ | 5.1 | 0.302 | 0.36 | 0.467 |

#### *B.3.2.7 Canopy/Windshield Drag Coefficient*

The drag coefficient due to the canopy and the windshield is assumed to be negligible in this research.

#### *B.3.2.8 Store Drag Coefficient*

It is assumed that there is no external storage attached to the vehicle and consequently the store drag coefficient will be reduced to zero.

#### *B.3.2.9 Trim Drag Coefficient*

The trim drag coefficient due to both the lift and the profile drag generated on the trimming surface is assumed to be negligible.

#### *B.3.2.10 Interference Drag Coefficient*

The interference drag originating from the fact that the drag of two components integrated in a single configuration is always larger than the sum of the individual drag components. The major interference drag factors have already been accounted for in the previous models: wing-fuselage, horizontal tail-vertical tail, empennage-fuselage, landing gear, nacelle-fuselage, and nacelle-wing. Any other contributors to the interference drag coefficient are thus neglected.

#### *B.3.2.11 Miscellaneous Drag Coefficient*

The contribution to the drag coefficient of the spoilers, the surface roughness, surface gaps, struts, antennas, and spillage are neglected in this research.

### B.3.3 General Transonic Drag Coefficient

Due to the lack of accurate analytical or numerical approaches for the transonic regime, Raymer [354] suggests an empirical approach to predict the drag coefficient between Mach 0.6 and Mach 1.2. This approach is based on several rules presented below:

- The drag coefficient at Mach 1.2 is equal to the drag coefficient at Mach 1.05.
- The drag coefficient at Mach 1 is half the drag coefficient at Mach 1.05.
- The drag coefficient has a smooth behavior over the entire transonic regime.
- The drag coefficient has a linear behavior between Mach 1 and Mach 1.05.

Following these rules, a parametric approach is developed using the aforementioned models to compute the vehicle's drag coefficient at Mach 0.6 and 1.2. Then, the transonic range is decomposed into three different ranges where different models are applied.

#### *B.3.3.1 From Mach 0.6 to Mach 1*

A second-order polynomial model is assumed for this range, as defined in Equation 245.

$$C_D = a_0 M^2 + b_0 M + c_0 \tag{245}$$

Using the same approach as the one implemented for the lift-curve slope, the matrix form of the problem to be solved is presented in Equation 246, where $C_D(1)$ is defined as the mean between $C_D(0.6)$ and $C_D(1.2)$. In addition, the minimum of that curve is assumed to be at Mach 0.6.

$$\begin{bmatrix} C_D(1) \\ C_D(0.6) \\ 0 \end{bmatrix} = \begin{bmatrix} 1^2 & 1 & 1 \\ 0.6^2 & 0.6 & 1 \\ 1.2 & 1 & 0 \end{bmatrix} \begin{bmatrix} a_0 \\ b_0 \\ c_0 \end{bmatrix} \tag{246}$$

The resolution of this system provides parametric values for $a_0$, $b_0$, and $c_0$, as described in Equation 247.

$$\begin{cases} a_0 &= 6.25\left(C_D(1) - C_D(0.6)\right) \\ b_0 &= 7.5\left(C_D(1) - C_D(0.6)\right) \\ c_0 &= 2.25 C_D(1) - 1.25 C_D(0.6) \end{cases} \tag{247}$$

#### *B.3.3.2 From Mach 1 to Mach 1.05*

As suggested by Raymer, a linear model is developed between Mach 1 and Mach 1.05, as presented in Equation 248.

$$C_D = 20M\left(C_D(1.2) - C_D(1)\right) + 21 C_D(1) - 20 C_D(1.2) \tag{248}$$

#### *B.3.3.3 From Mach 1.05 to Mach 1.2*

A second-order polynomial approximation is also selected for this range, as presented in Equation 249.

$$C_D = a_2 M^2 + b_2 M + c_2 \tag{249}$$

The corresponding system of equations to be solved is presented in Equation 250. In order to develop this system of equations, it is assumed that the maximum of the drag coefficient occurs at Mach 1.25, as suggested by Raymer. Hence, the curve has a symmetric behavior in this range.

$$\begin{bmatrix} C_D(1.2) \\ C_D(1.2) \\ 1.1C_D(1.2) \end{bmatrix} = \begin{bmatrix} 1.2^2 & 1.2 & 1 \\ 1.05^2 & 1.05 & 1 \\ 1.125^2 & 1.125 & 1 \end{bmatrix} \begin{bmatrix} a_2 \\ b_2 \\ c_2 \end{bmatrix} \tag{250}$$

The resolution of this system provides parametric values for $a_2$, $b_2$, and $c_2$ as described in Equation 251.

$$\begin{cases} a_2 &= -17.77C_D(1.2) \\ b_2 &= 40C_D(1.2) \\ c_2 &= -21.4C_D(1.2) \end{cases} \tag{251}$$

## *B.4 Propulsion Modeling*

### B.4.1 Rocket Engine Modeling

This section first provides the RSEs developed to evaluate the performance parameters of the selected propellants. Then, the actual vs. predicted plots as well as other key metrics used to check the goodness of fit are also presented.

#### *B.4.1.1 Response Surface Equations for Performance Estimation*

Equations 252 to 259 provide the RSEs developed to estimate the performance parameters of all selected rocket engines.

$$\text{Propellant C: } \begin{cases} Isp_v = 255.43 + 0.20 \log p_c + 15.54 \log \epsilon \\ c^* = 1569.53 + 9.34 \log p_c - 4.10^{-3} \log \epsilon \end{cases} \tag{252}$$

$$\text{O}_2/\text{H}_2\text{: } \begin{cases} Isp_v = 388.51 + 0.60 \log p_c + 17.09 \log \epsilon \\ c^* = 2460.58 + 6.39 \log p_c - 22.68 \log \epsilon \\ O/F = 2.86 + 0.086 \log p_c + 0.44 \log \epsilon \end{cases} \tag{253}$$

$$\text{O}_2/\text{RP1:} \begin{cases} Isp_v = 283.72 + 1.76 \log p_c + 18.59 \log \epsilon \\ c^* = 1784.90 + 17.13 \log p_c - 9.63 \log \epsilon \\ O/F = 2.20 + 0.05 \log p_c + 0.12 \log \epsilon \end{cases} \tag{254}$$

$$\text{N}_2\text{O}_4/\text{MMH:} \begin{cases} Isp_v = 279.41 + 0.91 \log p_c + 15.68 \log \epsilon \\ c^* = 1747.24 + 7.51 \log p_c - 10.72 \log \epsilon \\ O/F = 1.52 + 0.07 \log p_c + 0.15 \log \epsilon \end{cases} \tag{255}$$

$$\text{O}_2/\text{HTPB:} \begin{cases} Isp_v = 277.33 + 2.17 \log p_c + 18.49 \log \epsilon \\ c^* = 1751.93 + 17.40 \log p_c - 9.45 \log \epsilon \\ O/F = 1.93 + 0.05 \log p_c + 0.11 \log \epsilon \end{cases} \tag{256}$$

$$\text{O}_2/\text{Paraffin:} \begin{cases} Isp_v = 283.02 + 2.33 \log p_c + 18.59 \log \epsilon \\ c^* = 1787.97 + 16.72 \log p_c - 9.54 \log \epsilon \\ O/F = 2.24 + 0.05 \log p_c + 0.12 \log \epsilon \end{cases} \tag{257}$$

$$\text{N}_2\text{O}/\text{HTPB:} \begin{cases} Isp_v = 251.96 + 1.09 \log p_c + 13.57 \log \epsilon \\ c^* = 1769.52 + 12.13 \log p_c - 8.03 \log \epsilon \\ O/F = 6.22 + 0.09 \log p_c + 0.44 \log \epsilon \end{cases} \tag{258}$$

$$\text{N}_2\text{O}/\text{Paraffin:} \begin{cases} Isp_v = 255.14 + 0.50 \log p_c + 13.14 \log \epsilon \\ c^* = 1576.11 + 12.02 \log p_c - 7.11 \log \epsilon \\ O/F = 7.31 + 0.05 \log p_c + 0.42 \log \epsilon \end{cases} \tag{259}$$

#### *B.4.1.2 Regression Results*

The behavior of all actual vs. predicted plots are very similar so only one per type of propellant is provided in this section. They all benefit from good characteristics and are considered to be accurate enough for the purpose of this research, as displayed in Figures 188 to 190.

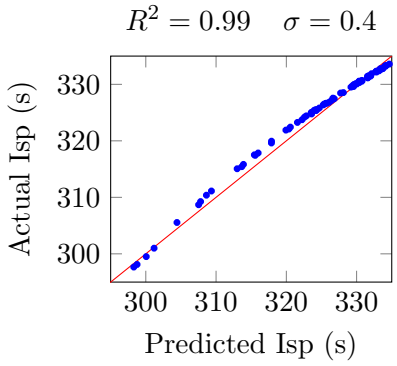



**Figure 188:** Actual vs. Predicted plot for Propellant C (solid)

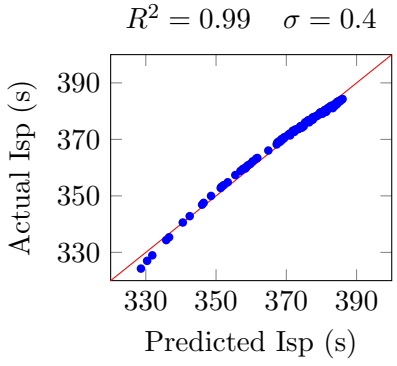



**Figure 189:** Actual vs. Predicted plot for $O_2$/RP1 (liquid)

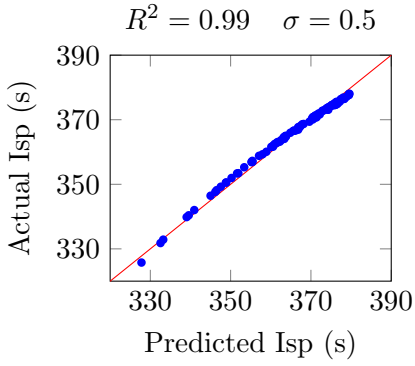



**Figure 190:** Actual vs. Predicted plot for $O_2$/HTPB (hybrid)

### B.4.2 Jet Engine Modeling

Jet engines are modeled based on the research conducted by Raymer [354]. In particular, Equations 260 to 267 provide a way to determine the jet engine dry weight $W_{jet}$, length

$L_{jet}$, diameter $D_{jet}$, and Specific Fuel Consumption (SFC) $SFC_{jet}$, respectively. These models are based on the available thrust $T_j$, the bypass ratio $BPR$, and the maximum Mach number $M$.

#### *B.4.2.1 Non-Afterburning Engines*

$$W_{jet} = 14.7 T_j^{1.1} \exp^{-0.045 BPR} \quad \text{(kg)} \tag{260}$$

$$L_{jet} = 0.49 T_j^{0.4} M^{0.2} \quad \text{(m)} \tag{261}$$

$$D_{jet} = 0.15 T_j^{0.5} \exp^{0.04 BPR} \quad \text{(m)} \tag{262}$$

$$SFC_{jet} = 19 \exp^{-0.12 BPR} \quad \text{(mg/Ns)} \tag{263}$$

#### *B.4.2.2 Afterburning Engines*

$$W_{jet} = 11.1 T_j^{1.1} M^{0.25} \exp^{-0.81 BPR} \quad \text{(kg)} \tag{264}$$

$$L_{jet} = 0.68 T_j^{0.4} M^{0.2} \quad \text{(m)} \tag{265}$$

$$D_{jet} = 0.11 T_j^{0.5} \exp^{0.04 BPR} \quad \text{(m)} \tag{266}$$

$$SFC_{jet} = 60 \exp^{-0.12 BPR} \quad \text{(mg/Ns)} \tag{267}$$

### *B.5 Description of the Cost Estimating Relationships*

This section describes the CERs provided by Goehlich [176] and used to evaluate the life-cycle costs of the airframe of all suborbital vehicles.

### B.5.1 Variable Description

The CERs provided in this section rely on the following variables:

- Project System Engineering Factor ($f_{PS}$): 1.075
- Technical Development Factor ($f_{TD}$): 1
- Technical Quality Factor:
  - Slender body ($f_{TQ,s}$): 1
  - Winged body ($f_{TQ,w}$): $M_m^{0.16}$, where $M_m$ is the maximum Mach number
- Team Experience Factor ($f_{TE}$): 1
- Integration Factor ($f_I$): 1.025
- Cost Reduction Factor ($f_{CR}$): 1
- Commercial Factor:
  - Rocket ($f_{C,r}$): 0.2
  - Jet ($f_{C,j}$): 1
  - Airframe ($f_{C,a}$): 0.5
- Cost Conversion Value ($d$): $-7.905 \times 10^{-6} y^2 + 37.31 \times 10^{-3} y - 42.78$ in which $y$ represents the fiscal year

In addition, $Mg$ is used as a baseline value for one tonne and $MY$ represents the average effort of one-year labor in the American aerospace industry.

### B.5.2 Development Cost

The development costs of a winged-body $C_{D,W}$ and a slender body $C_{D,B}$ are provided in Equations 268 and 269, respectively. In these equations, $M_W$ is the winged vehicle dry mass, $n_{R,V}$ the number of rocket engines, $M_R$ the dry mass of each rocket engine, $n_{J,V}$ the

number of jet engines, $M_J$ the dry mass of each jet engine, and $M_{0,B}$ the take-off gross weight of the vehicle.

$$C_{D,W} = 11,350 \frac{MY}{Mg^{0.335}} \left(M_W - n_{R,V} M_R - n_{J,V} M_J\right)^{0.335} f_{PS}.f_{TD,W}.f_{TQ,W}.f_{TE,W}.f_{CD,W}.d \tag{268}$$

$$C_{D,B} = \left(42 \frac{MY}{Mg}.M_{0,B} + 30000MY\right).f_{CD,B}.d \tag{269}$$

### B.5.3 Manufacturing Cost

The manufacturing costs of a winged-body $C_{V,W}$ and a slender body $C_{V,B}$ are provided in Equations 271 and 270, respectively. In these equations, $M_W$ is the winged vehicle dry mass, $M_{0,B}$ the dry mass of the vehicle, $n_{R,V}$ the number of rocket engines, $M_R$ the dry mass of each rocket engine, $n_{J,V}$ the number of jet engines, and $M_J$ the dry mass of each jet engine.

$$C_{V,W} = 84.3 \frac{MY}{Mg^{0.669}} M_W^{0.669}.f_I^N.f_{CR,W}.f_{CP,W}.d \tag{270}$$

$$C_{V,B} = 638.6 \frac{MY}{Mg^{0.485}} \left(M_B - n_{R,V} M_R - n_{J,V} M_J\right)^{0.485} f_I^N.f_{CR,B}.f_{CP,B}.d \tag{271}$$

### B.5.4 Variable Direct Operating Cost

All variable operating cost components are provided in Equations 272 to 279: the pre-launch operating cost for horizontal launch $C_{PL,H}$, the pre-launch operating cost for vertical launch $C_{PL,V}$, the launch operating cost $C_{LO}$, the launch site cost $C_{LS}$, the vehicle insurance cost $C_{Ins}$, the vehicle amortization cost $C_{VA}$, the transportation cost $C_T$, and the airframe maintenance cost $C_M$.

In these equations, $L$ is the number of launches per year for the total fleet, $M_0$ the total vehicle launch mass, $T_m$ the mission duration, $n_c$ the number of crew members, $n_p$ the number of passengers, $R_{abort}$ the insurance abort rate, $TOC$ the total operating cost, $R_{loss}$

the insurance loss rate, $C_v$ the vehicle manufacturing cost, $r_{B/W}$ the number of reuses of the airframe, $r_R$ the number of reuses of the rocket engines, $r_J$ the number of reuses of the jet engines, $C_{V,B/W}$ the airframe manufacturing cost, $C_{V,R}$ the rocket engine manufacturing cost, $C_{V,J}$ the jet engine manufacturing cost, and $M_v$ the vehicle dry mass.

$$C_{PL,H} = 1.409 \frac{MY}{launch^{0.46}.year^{0.54}.Mg^{0.75}} L^{-0.54} M_0^{0.75} d \tag{272}$$

$$C_{PL,V} = 6.618 \frac{MY}{launch^{0.39}.year^{0.61}.Mg^{0.75}} L^{-0.61} M_0^{0.75} d \tag{273}$$

$$C_{LO} = \left( 20 \frac{launch^{-0.35}}{year^{0.65}} L^{-0.65} + 0.42 \frac{launch^{-0.2}}{hour.year^{0.8}} T_m \left( n_c + n_p \right)^{0.5} \right) d \tag{274}$$

$$C_{LS} = 0.1. \frac{MY}{launch}.d \tag{275}$$

$$C_{Ins} = R_{abort} TOC.launch + R_{loss} C_V \tag{276}$$

$$C_{VA} = \left( \frac{1}{r_{B/W}}.C_{V,B/W} + \frac{1}{r_R}.C_{V,R} + \frac{1}{r_J}.C_{V,J} \right) . \frac{1}{launch} \tag{277}$$

$$C_T = 72.1 \times 10^{-3} \frac{MY}{Mg^{0.5}.launch}.M_v^{0.5} d \tag{278}$$

$$C_M = \begin{cases} 0.004 C_{V,B/W} \cdot \frac{1}{launch} & \text{for single stage vehicles} \\ 0.0001 C_{V,B/W} \cdot \frac{1}{launch} & \text{for second stage vehicles} \end{cases} \tag{279}$$

### B.5.5 Fixed Direct Operating Cost

All fixed direct operating cost components are provided in Equations 280 to 283: the development amortization cost $C_{DA}$, the financing cost $C_F$, the product improvement cost $C_{PI}$, and the abolition cost $C_{AB}$.

In these equations, $C_D$ is the total development cost, $IR$ the interest rate, $T_{IR}$ the years of the interest rate, $RR$ the repayment rate, $T_{RR}$ the years of the repayment rate, $L$

the number of launches per year, $U$ the duration of the program, $n_{B/W,F}$ the number of produced vehicle, and $DOC_{var}$ the variable direct operating cost.

$$C_{DA} = \frac{1}{L.U}.C_D \tag{280}$$

$$C_F = C_V \left( (1+IR)^{T_{IR}} \frac{RR}{1-(1+RR)^{-T_{RR}}} T_{RR} - 1 \right) \frac{1}{L.U} \tag{281}$$

$$C_{PI} = 0.045 \frac{1}{year^{0.33}} \frac{1}{L.U} \left( n_{B/W,F}.U \right)^{0.33} C_D \tag{282}$$

$$C_{AB} = \frac{n_V}{L.U}.DOC_{var}.launch \tag{283}$$

### B.5.6 Indirect Operating Cost

The indirect operating cost is directly represented by the administration cost $C_A$, as described in Equation 284.

$$C_A = 0.9 \frac{MY}{launch} d \tag{284}$$

# APPENDIX C

# DEVELOPMENT OF THE VISUALIZATION PLATFORM

This section aims at describing the model developed for each component of the vehicle using Catia.

#### *C.0.6.1 Propulsion Systems*

**Nozzle:** In order to make the CAD simpler, no distinction is made between the nozzle used with solid, liquid, and hybrid rocket engines. The geometric shape of the nozzle is based on a bell-shaped rocket engine and cooling tubes are added on the external surface of both the nozzle and the combustion chamber, as displayed in Figure 191. The nozzle model is driven by three parameters: the combustion chamber diameter, the nozzle length, and the external nozzle diameter.

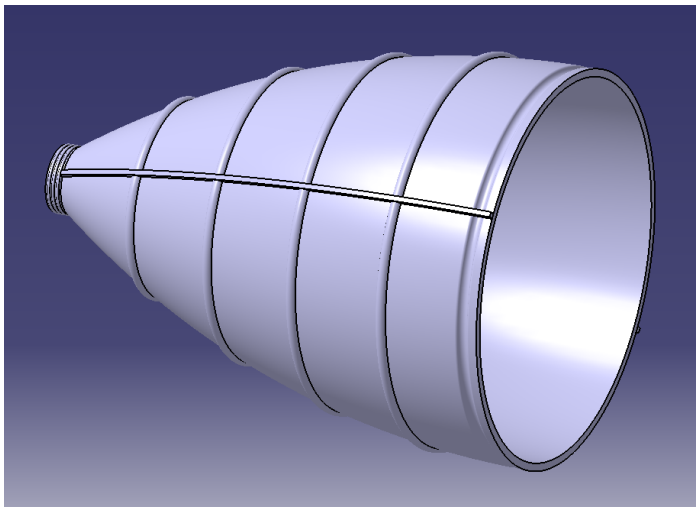


**Figure 191:** Model of the rocket engine nozzle

**Propellant tank:** The propellant tank section can be modeled by three technological solutions based on the type of propellant used: solid, liquid, and hybrid. Each solution is displayed or hidden using a Boolean parameter defined by the selected architecture. The

different models are discussed below:

- Solid engine model: the model of the solid engine is composed of the propellant grain and the corresponding fuselage section. The propellant grain is directly linked to the fuselage wall and the thermal protection thickness is neglected. A "star" pattern hole is chosen for the engine port shape, as described in Figure 192. The solid engine module is driven by two parameters: the vehicle fuselage diameter and the solid engine length.

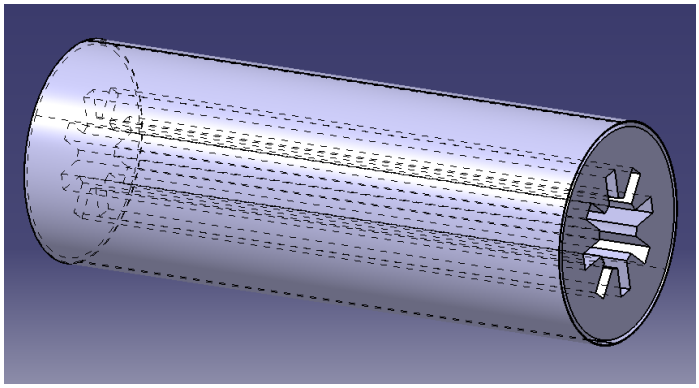


**Figure 192:** Model of the solid engine

- Liquid engine model: the liquid engine is composed of a fuel tank, an oxidizer tank, and a section of the fuselage, as presented in Figure 193. The model is driven by 5 parameters: the fuel tank diameter and length, the oxidizer tank diameter and length, and the diameter of the fuselage.

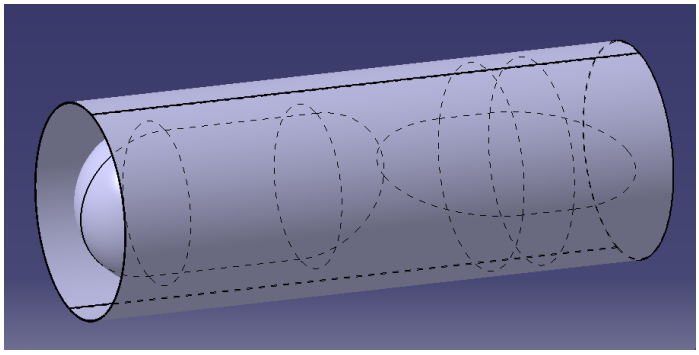


**Figure 193:** Model of the liquid engine tanks

- Hybrid engine model: the hybrid engine is composed of both liquid and solid propellants. Hence, the geometric shapes described in the previous sections are used to model the hybrid engine, as shown in Figure 194. The model is driven by five parameters: the oxidizer tank diameter and length, the solid engine diameter and length, and the diameter of the fuselage.

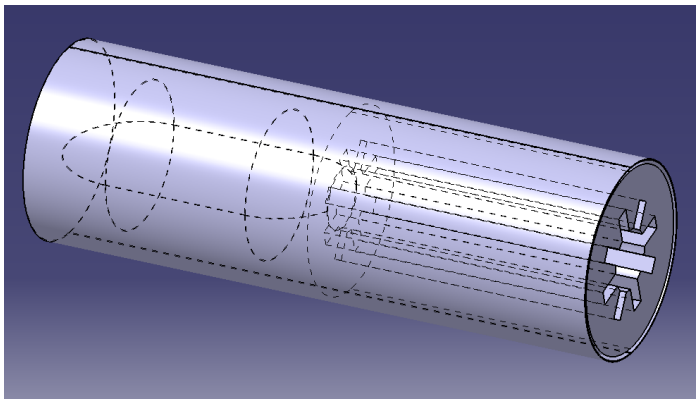


**Figure 194:** Model of the hybrid engine tanks

**Jet engines:** Two models of jet engine configurations are considered in this design: side jet engines (directly linked to the fuselage) and central jet engines (integrated into structure of the vertical tail) when an odd number of jet engines is selected:

- Side jet engines: to display each potential architecture using Boolean parameters, four different side jet engines are created. Each part is composed of four components: the engine body, the inlet fan with blades, the outlet nozzle, and the mechanical junction between the engine body and the fuselage of the vehicle, as displayed in Figure 195. The shape of the side engines is determined by three parameters: the inside engine diameter, the engine length, and the fuselage diameter.

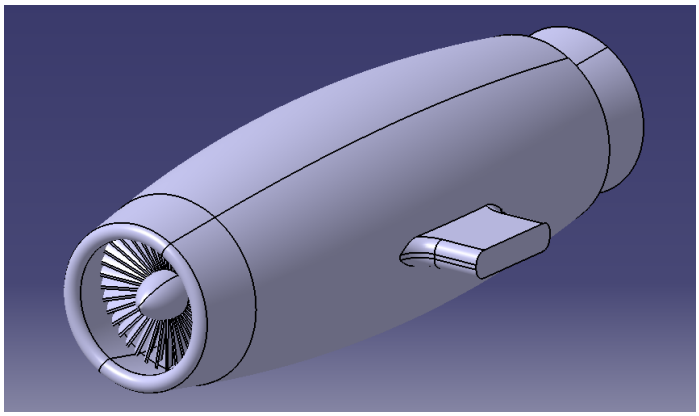


**Figure 195:** Model of the side engines

- Central jet engines: the central jet engine has to be integrated into the vertical tail. In order to have the nozzle behind the trailing edge of the vertical tail, the engine is extended, adding a new parameter: the vertical tail root chord, as presented in Figure 196.

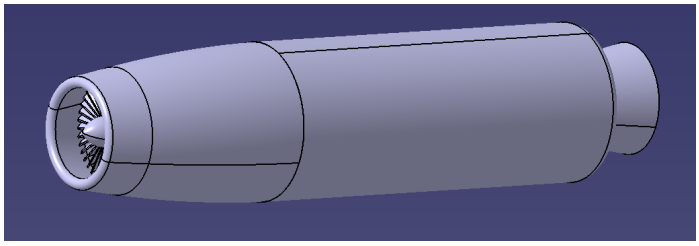


**Figure 196:** Model of the central jet engines

### C.0.7 Lifting and Control Surfaces

For winged vehicles, three different lifting/control surfaces must be considered: wing, vertical tail, and horizontal tail. All these surfaces are assumed to have a trapezoidal shape defined by the same parameters: the root chord, the tip chord, the span, and the thickness-to-chord ratio. The design of such surfaces is done following a three-step process. First, a shape of the airfoil is modeled to generate both the root and the tip airfoils. Then, a surface is created that links the two airfoils, and finally, the volumetric body part is generated. If necessary, a symmetric geometry is created.

To model the airfoil, the symmetric four-digit NACA airfoil developed by the National

Advisory Committee for Aeronautics (NACA) is chosen and modified with a parametric thickness-to-chord ratio. An airfoil shape surface is created with Catia's generative shape design module using Equation 285, where $t_c$ is the thickness-to-chord ratio, $x$ the x-coordinate, and $c$ the chord. The modeling of the airfoil is based on a discretization of the airfoil into 20 points, as shown in Figure 197.

$$y_{t_c}(x) = 5t_c c\left(0.2969\sqrt{\frac{x}{c}} - 0.1260\left(\frac{x}{c}\right) - 0.3516\left(\frac{x}{c}\right)^2 + 0.2843\left(\frac{x}{c}\right)^3 - 0.1015\left(\frac{x}{c}\right)^4\right) \tag{285}$$

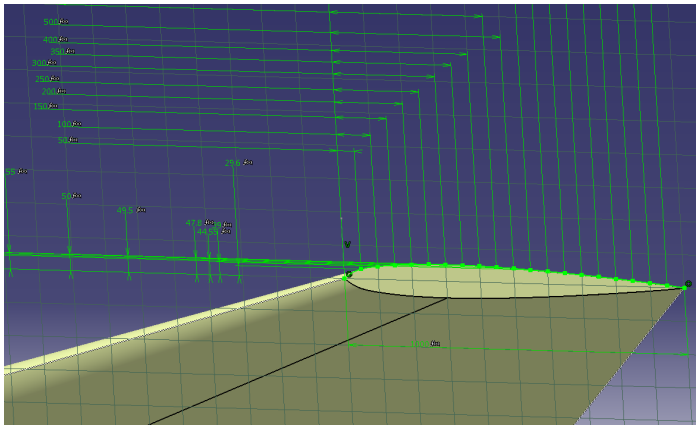


**Figure 197:** Symmetric four-digit NACA airfoil model

Once the different surfaces have been created, a volumetric body part is generated for each lifting surface model: wing, vertical tail, and horizontal tail. These parts are described in the following sections.

*C.0.7.1 Wing*

A symmetry is performed on the previously generated half wing. An additional part is designed to smoothly link the wing to the fuselage, as displayed in Figure 198. Two more parameters are added to the ones inherent to all lifting surfaces: the fuselage diameter and the length between the wing trailing edge and the back of the fuselage in order to locate the wing along the horizontal axis.

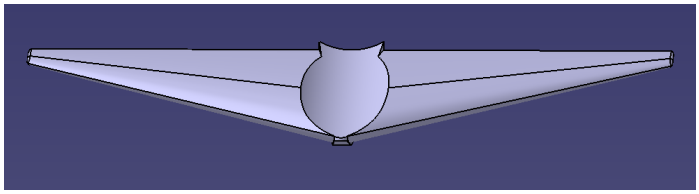


**Figure 198:** Wing model with junction

*C.0.7.2 Vertical and Horizontal Tails*

In order to model the empennage, a symmetric horizontal tail is attached to the non-symmetric vertical tail, as shown in Figure 199. In order to fix the height of the horizontal tail, a constraint is applied that requires the horizontal tail root chord to be equal to the vertical tail chord at the junction. Similar to the wing, two parameters are added: the junction height and the fuselage diameter.

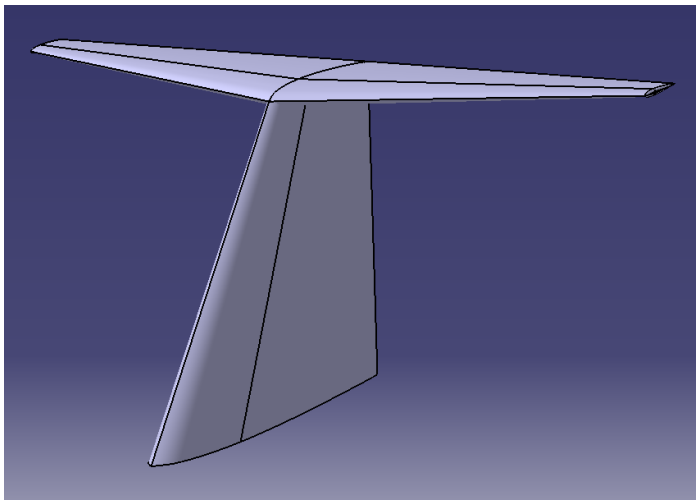


**Figure 199:** Empennage model

### C.0.8 Fuselage

The fuselage is composed of three sections: the equipment bay, the cabin, and the cockpit. The modeling of each of these sections is discussed below:

- Equipment bay: the equipment section contains all electrical, hydraulic, and pneumatic systems. It is located above the vehicle floor and its size is assumed to be fixed, as displayed in Figure 200. Hence, the only parameter is the fuselage diameter.

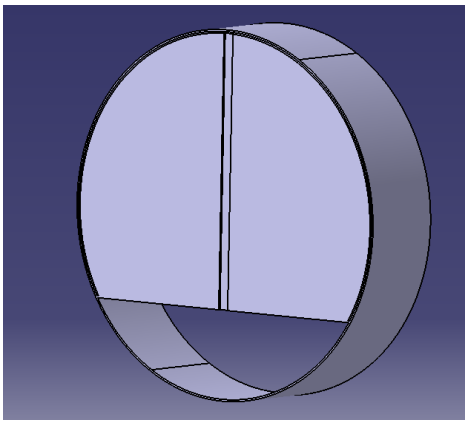
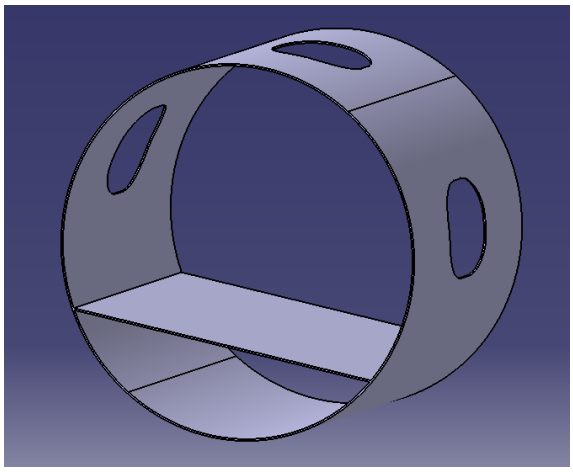


**Figure 200:** Model of the equipment bay

- Cabin: the cabin corresponds to the habitable and is composed of the passengers sections, the seats, and the cockpit:
  - Passenger sections: passenger sections are decomposed into the fuselage part with large observation windows (which allow passengers to see the Earth from space, making their flight an incredible experience), and the vehicle floor, as described in Figure 201(a). The vehicle can carry up to four passenger sections with a full capacity of eight passengers. Each passenger section is driven by two parameters: the fuselage diameter, and the seat pitch. The latter defines the passenger available space to enjoy the weightlessness phase, which is representative of the passenger comfort.
  - Seats: as displayed in Figure 201(b), seats are designed to make the passenger experience as comfortable as possible. In order to reduce the acceleration effects on the passenger body, passenger seats can also take a lie down position. The vehicle can carry up to two pilot seats and eight passenger seats. The dimensions of those seats are fixed, so only Boolean parameters drive them.



(a) Model of the passenger section

(b) Model of the seats

– Cockpit: the cockpit has two versions: a capsule version used for slender bodies, and a aircraft version for winged-body configurations. Due to the complexity of the geometrical shapes, only the vehicle fuselage diameter varies for these parts. Both versions are discussed below:

  * Capsule: the capsule is composed of a conical shape structure and a floor. A trapdoor, attitude thrusters, and two small windows are added to the design, as displayed in Figure 201.

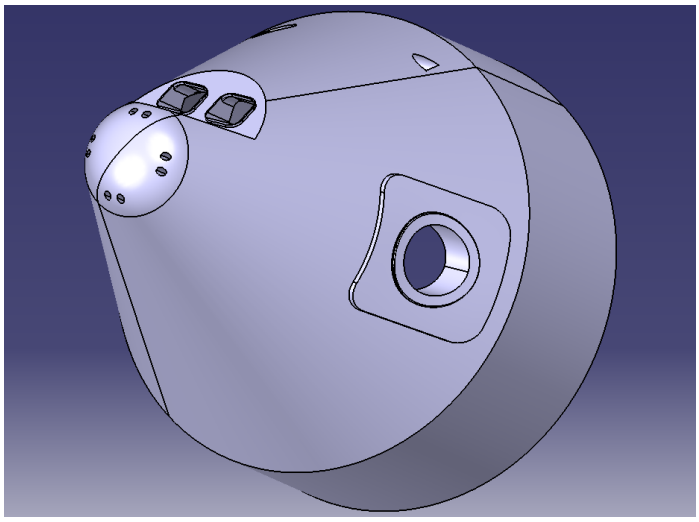


**Figure 201:** Model of the capsule cockpit

* Cockpit: the aircraft cockpit is composed of an external shape and a floor. Due to its complexity, the external shape is created with the Catia's generative shape design module, and then transformed into volumetric parts, as illustrated in Figure 202.

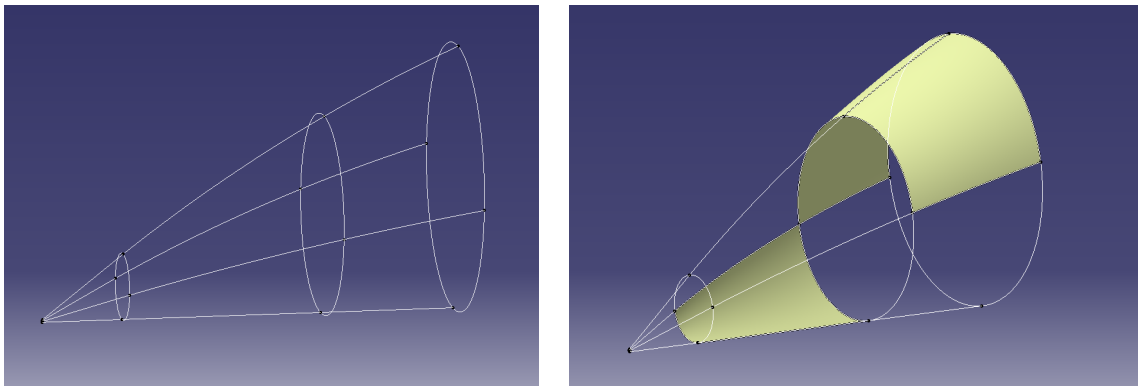


**Figure 202:** Development of the aircraft cockpit model

Then, all surfaces that compose the aircraft cockpit are joined and transformed into a volumetric part body. Large windows are added to the design in order to help pilots during the landing phase, as shown in Figure 203.

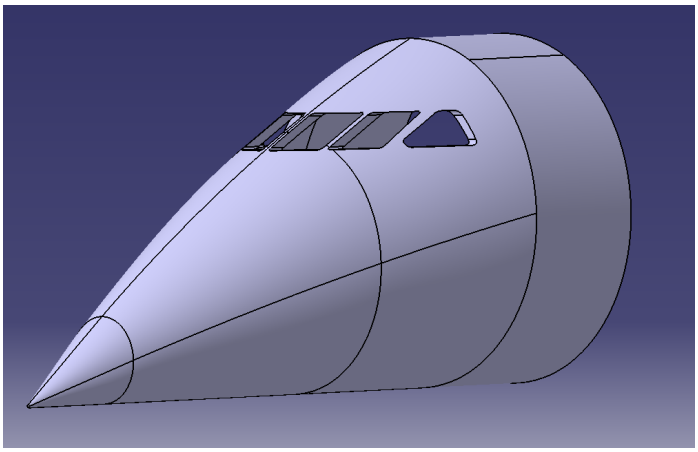


**Figure 203:** Model of the aircraft cockpit

# APPENDIX D

# MATLAB CODE

## *D.1 Implementation of ENVISAGE*

### D.1.1 Architecture of the Matlab Program

The architecture of the software is presented in Figure 204. It is composed of 23 functions, whose 12 are linked to user interfaces (gray boxes). This section describes the main functions that are linked to the user interface and their relationships with the other functions.

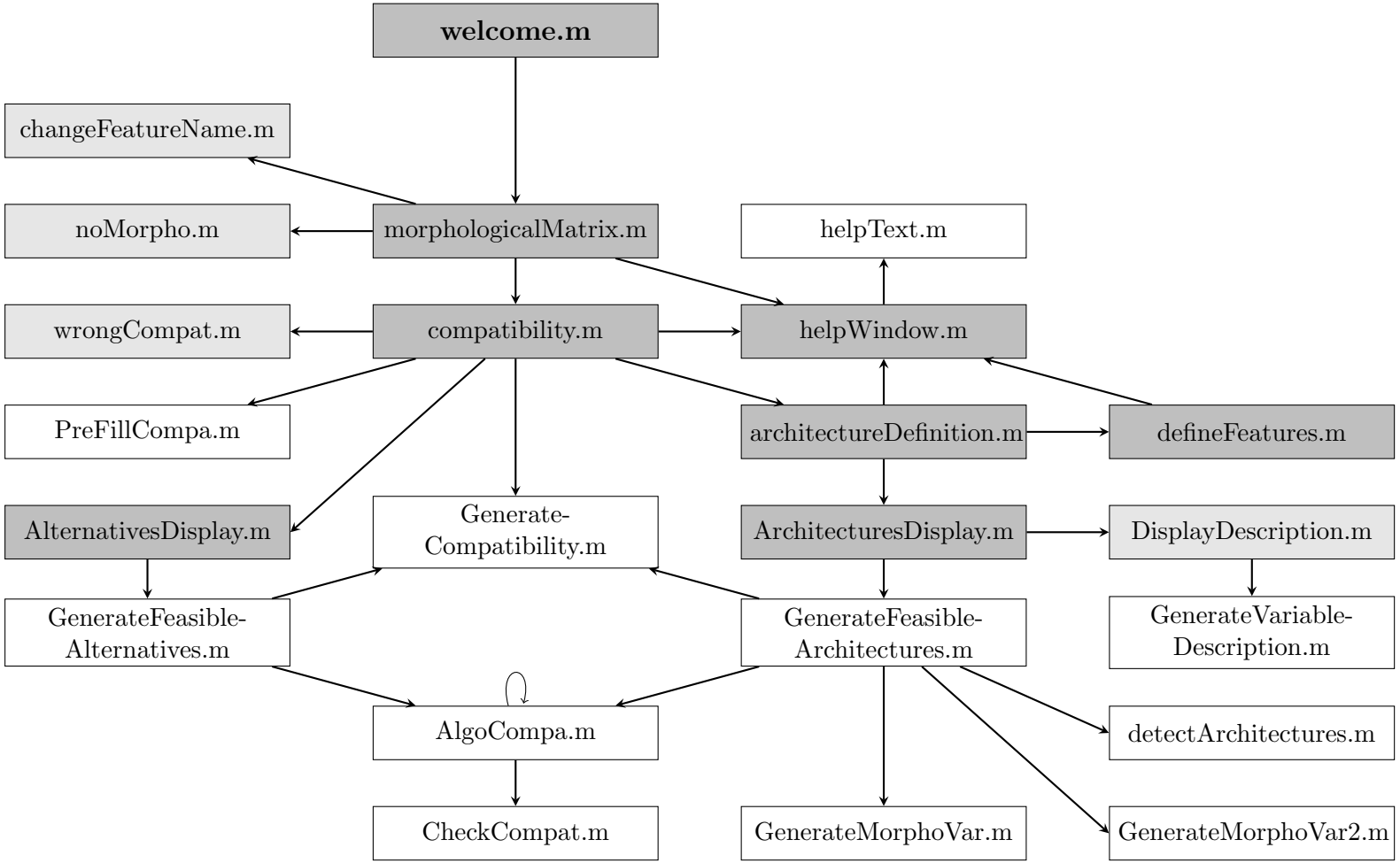



**Figure 204:** Architecture of the software ENVISAGE

### D.1.2 Matlab Functions

#### *D.1.2.1 welcome.m*

```
function varargout = welcome(varargin)
%Function that opens the welcom window

%Initialisation of the window
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @welcome_OpeningFcn, ...
                   'gui_OutputFcn',  @welcome_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end

function welcome_OpeningFcn(hObject, eventdata, handles, varargin)
%Opening function (executes just before welcome is made visible)

%Display asdl logo
imshow('asdl.png')

%Choose default command line output for welcome
handles.output = hObject;

%Update handles structure
guidata(hObject, handles);

%UIWAIT makes welcome wait for user response
 uiwait(handles.figure1);

function varargout = welcome_OutputFcn(hObject, eventdata, handles)
%Output fucntion (returns what is needed)

function pushbutton1_Callback(hObject, eventdata, handles)
%Start button (open next window)
morphologicalMatrix(handles.figure1);

function figure1_CloseRequestFcn(hObject, eventdata, handles)
```

```
%Close request function (executes when user attempts to close figure1)
delete(hObject);
```

#### *D.1.2.2 morphologicalMatrix.m*

```
function varargout = morphologicalMatrix(varargin)
%Definition of the morphological matrix

% Begin initialization code
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
    'gui_Singleton',  gui_Singleton, ...
    'gui_OpeningFcn', @morphologicalMatrix_OpeningFcn, ...
    'gui_OutputFcn',  @morphologicalMatrix_OutputFcn, ...
    'gui_LayoutFcn',  [] , ...
    'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end

function morphologicalMatrix_OpeningFcn(hObject, eventdata, handles,...
    varargin)
%Opening function (executes just before welcome is made visible)

%Close previous window
delete(varargin{1});

%Display picture
imshow('logo.png')

%Define column names
handles.columnNames={'Option 1','Option 2'};
set(handles.uitable3, 'columnname', handles.columnNames);

%Define row names
handles.rowNames={'Feature 1','Feature 2'};
set(handles.uitable3, 'rowname', handles.rowNames);


set(handles.popupmenu2,'String', handles.rowNames);
```

```
42
43 % Choose default command line output for test
44 handles.output = hObject;
45
46 % Update handles structure
47 guidata(hObject, handles);
48
49 % UIWAIT makes test wait for user response (see UIRESUME)
50 uiwait(handles.figure1);
51
52 function varargout = morphologicalMatrix_OutputFcn(hObject, eventdata,...
53     handles)
54 %Output fucntion (returns what is needed)
55
56 function pushbutton3_Callback(hObject, eventdata, handles)
57 %Add features to the matrix
58 data=get(handles.uitable3, 'data');
59 newline=cell(1,size(data,2));
60 newline(:)={''};
61 data=cat(1,data,newline);
62 handles.strHeadersRow=cat(2, 'Feature ',num2str(size(data,1)));
63 handles.rowNames=get(handles.uitable3,'rowname');
64 handles.rowNames=cat(2,handles.rowNames',handles.strHeadersRow);
65 set(handles.uitable3,'data', data);
66 set(handles.uitable3, 'rowname', handles.rowNames);
67
68 %Update popup menu
69 set(handles.popupmenu2,'String', handles.rowNames);
70
71 %Update handles structure
72 guidata(hObject, handles);
73
74 function pushbutton4_Callback(hObject, eventdata, handles)
75 %Add options to the matrix
76 data=get(handles.uitable3, 'data');
77 newcolumn=cell(size(data,1),1);
78 newcolumn(:)={''};
79 data=cat(2,data,newcolumn);
80 handles.columnNames=get(handles.uitable3,'columnname');
81 handles.strHeaders=cat(2, 'Option ',num2str(size(data,2)));
82 handles.columnNames=cat(2,handles.columnNames',handles.strHeaders);
83 set(handles.uitable3,'data', data);
84 set(handles.uitable3, 'columnname', handles.columnNames);
85 guidata(hObject, handles);
86
87 function pushbutton6_Callback(hObject, eventdata, handles)
88 %Ensure compatibility between options: open a new window and close this one
89 varargout{1} = get(handles.uitable3, 'data');
90 MMatrix=get(handles.uitable3, 'data');
```

```
emptyCells = cellfun(@isempty,MMatrix);
MMatrix(emptyCells) = [];

%Check if morpho is empty
if length(MMatrix)==0
    noMorpho(); %Error message
else
    handles.columnNames=get(handles.uitable3,'columnname');
    handles.rowNames=get(handles.uitable3,'rowname');
    compatibility(varargout{1},handles.figure1,handles.rowNames,...
        handles.columnNames); %Executes compatibility.m
end

function figure1_CloseRequestFcn(hObject, eventdata, handles)
%Executes when user attempts to close figure1.
delete(hObject);

function uipushtool1_ClickedCallback(hObject, eventdata, handles)
%Load existing morphologocial matrix
warning('off','MATLAB:table:ModifiedVarnames');
possiblefile={'*.xlsx';'*.xls'};
[FileName,PathName] = uigetfile(possiblefile,'Select an Excel file');

if FileName == 0

else
    filePathName=cat(2,PathName,FileName);
    T=readtable(filePathName);
    Tdata1=table2cell(T);
    Tdata=Tdata1(:,[2:size(T,2)]);

    for i=1:size(Tdata,2)
        columnHead{i}= cat(2, 'Option ',num2str(i));
    end
    set(handles.uitable3,'data',Tdata);
    set(handles.uitable3, 'rowname', T{:,1});
    set(handles.uitable3, 'columnname', columnHead);
    handles.rowNames= T{:,1};
end

%Update popup menu
set(handles.popupmenu2,'String', handles.rowNames);
guidata(hObject, handles);

function uipushtool2_ClickedCallback(hObject, eventdata, handles)
%Save morphological matrix in Excel
possiblefile={'*.xlsx';'*.xls'};
[FileName,PathName] = uiputfile(possiblefile);

```

```
if FileName == 0

else
    filePathName=cat(2,PathName,FileName);
    headers=get(handles.uitable3, 'columnname');
    dataToBeWritten=get(handles.uitable3, 'data');
    oldNames=get(handles.uitable3, 'rowname');
    rowHeader=cat(2,' ',oldNames');
    matrixToBeWritten=cat(1,headers',dataToBeWritten);
    matrixToBeWritten=cat(2,rowHeader',matrixToBeWritten);
    a = dir(PathName);
    b = struct2cell(a);
    if any(ismember(b(1,:),FileName)) == 1
        EraseExcelSheets(filePathName);
    end
    xlswrite(filePathName,matrixToBeWritten);
end

function popupmenu2_Callback(hObject, eventdata, handles)
%Executes on selection change in popupmenu2.
val = get(hObject,'Value');
name=get(handles.uitable3,'rowname');
outputtt=changeFeatureName(name((val)));
name(val)={outputtt{1}};
set(handles.uitable3,'rowname',name);
set(handles.popupmenu2,'String', get(handles.uitable3,'rowname'));
guidata(hObject,handles);

function popupmenu2_CreateFcn(hObject, eventdata, handles)
%Executes during creation of the popupmenu2
if ispc && isequal(get(hObject,'BackgroundColor'),...
        get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function uipushtool4_ClickedCallback(hObject, eventdata, handles)
%Help window
helpWindow('morpho');
```

#### *D.1.2.3 compatibility.m*

```
function varargout = compatibility(varargin)
%Display compatibility matrix that allows users to define compatibility
%between all features

%Begin initialization code
```

```
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
    'gui_Singleton',  gui_Singleton, ...
    'gui_OpeningFcn', @compatibility_OpeningFcn, ...
    'gui_OutputFcn',  @compatibility_OutputFcn, ...
    'gui_LayoutFcn',  [] , ...
    'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end


function compatibility_OpeningFcn(hObject, eventdata, handles, varargin)
%Executes just before compatibility is made visible.

%Close previous window
delete(varargin{2});

%Display picture
imshow('logo.png')

%Generate empty compatibility matrix
handles.uitable2=[];
handles.Morpho=varargin{1};
handles.rowNames=varargin{3};
handles.columnNames=varargin{4};
listOptions=GenerateCompatibility(varargin{1},varargin{1});
handles.listOp=listOptions;

%Prefill compatibility matrix with 0 for options from the same feature
handles.sizeCompat=length(listOptions);
handles.uitable2=cell(handles.sizeCompat,handles.sizeCompat);
handles.uitable2(:)={''};
handles.uitable2 = PreFillCompa(handles.uitable2,varargin{1});
data=get(handles.uitable1, 'data');
set(handles.uitable1,'data', handles.uitable2);

%Row title
set(handles.uitable1, 'rowname', listOptions);
%Column title
set(handles.uitable1, 'columnname', listOptions);

%Choose default command line output for compatibility
```

```
55 handles.output = hObject;
56
57 %Update handles structure
58 guidata(hObject, handles);
59
60 %UIWAIT makes compatibility wait for user response
61 uiwait(handles.figure1);
62
63 function varargout = compatibility_OutputFcn(hObject, eventdata, handles)
64 %Outputs from this function are returned to the command line.
65
66 function pushbutton1_Callback(hObject, eventdata, handles)
67 %Executes on button press in "Generate Alternatives".
68 varargout{1} = get(handles.uitable1, 'data');
69 AlternativesDisplay(handles.Morpho, varargout{1}, handles.figure1,...
70     handles.rowNames);
71
72 function figure1_CloseRequestFcn(hObject, eventdata, handles)
73 %Executes when user attempts to close figure1.
74 delete(handles.figure1);
75
76
77 function uipushtool1_ClickedCallback(hObject, eventdata, handles)
78 %Load compatibility matrix
79 warning('off','MATLAB:table:ModifiedVarnames');
80 possiblefile={'*.xlsx';'*.xls'};
81 [FileName,PathName] = uigetfile(possiblefile,'Select an Excel file');
82
83 if FileName == 0
84
85 else
86     filePathName=cat(2,PathName,FileName);
87     T=readtable(filePathName);
88     Tdata1=table2cell(T);
89     Tdata=Tdata1(:,[2:size(T,2)]);
90
91     %Transform into a cell of strings
92     d=size(Tdata);
93     for i=1:d(1)
94         for j=1:d(2)
95             a=cell2mat(Tdata(i,j));
96             if a==0 && i<=j
97                 Tdata{i,j}='0';
98             elseif a==1 && i<=j
99                 Tdata{i,j}='1';
100            else
101                Tdata{i,j}='';
102            end
103        end
```

```
104     end
105
106     %Test dimensions of the compatibility matrix compared to morphological.
107     %If ok proceed, otherwise error message
108     if size(Tdata,1)==handles.sizeCompat && ...
109             size(Tdata,2) == handles.sizeCompat
110         set(handles.uitable1,'data',Tdata);
111         set(handles.uitable1, 'rowname', T{:,1});
112         set(handles.uitable1, 'columnname', T{:,1});
113         guidata(hObject, handles);
114     else
115         wrongCompat();
116     end
117 end
118
119 function uipushtool2_ClickedCallback(hObject, eventdata, handles)
120 %Save compatibility matrix
121 possiblefile={'*.xlsx';'*.xls'};
122 [FileName,PathName] = uiputfile(possiblefile);
123
124 if FileName == 0
125
126 else
127     filePathName=cat(2,PathName,FileName);
128     headers=get(handles.uitable1, 'columnname');
129     dataToBeWritten=get(handles.uitable1, 'data');
130     oldNames=get(handles.uitable1, 'rowname');
131     rowHeader=cat(2,' ',oldNames');
132     matrixToBeWritten=cat(1,headers',dataToBeWritten);
133     matrixToBeWritten=cat(2,rowHeader',matrixToBeWritten);
134     a = dir(PathName);
135     b = struct2cell(a);
136     if any(ismember(b(1,:),FileName)) == 1
137         EraseExcelSheets(filePathName);
138     end
139     xlswrite(filePathName,matrixToBeWritten);
140 end
141
142 function pushbutton2_Callback(hObject, eventdata, handles)
143 %Executes on button press in pushbutton2.
144 architectureDefinition(handles.Morpho,handles.rowNames,...
145     handles.columnNames,handles.listOp, handles.uitable1, handles.figure1);
146
147 function uipushtool4_ClickedCallback(hObject, eventdata, handles)
148 %Executes when pressing the help button
149 helpWindow('compat');
```

#### *D.1.2.4 AlternativesDisplay.m*

```
function varargout = AlternativesDisplay(varargin)
%Display list of feasible alternatives

%Begin initialization code
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
    'gui_Singleton',  gui_Singleton, ...
    'gui_OpeningFcn', @AlternativesDisplay_OpeningFcn, ...
    'gui_OutputFcn',  @AlternativesDisplay_OutputFcn, ...
    'gui_LayoutFcn',  [] , ...
    'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end

function AlternativesDisplay_OpeningFcn(hObject,eventdata,handles,varargin)
%Executes just before AlternativesDisplay is made visible.

%Display picture
imshow('logo.png')

%Create table
MorphoTest=varargin{1};
CompatMatrix1=varargin{2};
handles.rowNames=varargin{4};
d=size(CompatMatrix1);
for i=1:d(1)
    for j=1:d(2)
        a=cell2mat(CompatMatrix1(i,j));
        CompatMatrix{i,j}=num2str(a);
    end
end

%Generate feasible alternatives
temp=GenerateFeasibleAlternatives(MorphoTest,CompatMatrix);
handles.ListFeasibleAlternatives=temp{1};
numOp=temp{2};

%Fill table
set(handles.uitable1,'data', handles.ListFeasibleAlternatives);
```

```
47 set(handles.uitable1, 'columnname',handles.rowNames);
48
49 %Write total number of alternatives
50 textToBeWritten1='Number of feasible alternatives: ';
51 textToBeWritten2='Number of possible alternatives: ';
52 textToBeWrittenTop=cat(2,textToBeWritten1,...
53     num2str(size(handles.ListFeasibleAlternatives,1)));
54 textToBeWrittenDown=cat(2,textToBeWritten2,num2str(numOp));
55 set(handles.text2, 'String', textToBeWrittenTop);
56 set(handles.text3, 'String', textToBeWrittenDown);
57
58 % Choose default command line output for AlternativesDisplay
59 handles.output = hObject;
60
61 % Update handles structure
62 guidata(hObject, handles);
63
64 % UIWAIT makes AlternativesDisplay wait for user response
65 uiwait(handles.figure1);
66
67 function varargout = AlternativesDisplay_OutputFcn(hObject,...
68     eventdata, handles)
69 %Outputs from this function are returned to the command line.
70
71 function uipushtool2_ClickedCallback(hObject, eventdata, handles)
72 %Save lsit of feasible alternatives
73 possiblefile={'*.xlsx';'*.xls'};
74 [FileName,PathName] = uiputfile(possiblefile);
75
76 if FileName == 0
77
78 else
79     filePathName=cat(2,PathName,FileName);
80     headers=get(handles.uitable1, 'columnname');
81     dataToBeWritten=get(handles.uitable1, 'data');
82     rowHeaders=[];
83     rowHeaders{1}='';
84     for i=1:size(dataToBeWritten,1)
85         rowHeaders{i+1}=num2str(i);
86     end
87     matrixToBeWritten=cat(1,headers',dataToBeWritten);
88     matrixToBeWritten=cat(2,rowHeaders',matrixToBeWritten);
89     a = dir(PathName);
90     b = struct2cell(a);
91     if any(ismember(b(1,:),FileName)) == 1
92         EraseExcelSheets(filePathName);
93     end
94     xlswrite(filePathName,matrixToBeWritten);
95 end
```

```
96
97 function figure1_CloseRequestFcn(hObject, eventdata, handles)
98 %Executes when user attempts to close figure1.
99 delete(handles.figure1);
```

#### *D.1.2.5 helpWindow.m*

```
1  function varargout = helpWindow(varargin)
2  %Display the help window as a function of the parent window
3
4  % Begin initialization code
5  gui_Singleton = 1;
6  gui_State = struct('gui_Name',       mfilename, ...
7                     'gui_Singleton',  gui_Singleton, ...
8                     'gui_OpeningFcn', @helpWindow_OpeningFcn, ...
9                     'gui_OutputFcn',  @helpWindow_OutputFcn, ...
10                    'gui_LayoutFcn',  [] , ...
11                    'gui_Callback',   []);
12 if nargin && ischar(varargin{1})
13     gui_State.gui_Callback = str2func(varargin{1});
14 end
15
16 if nargout
17     [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
18 else
19     gui_mainfcn(gui_State, varargin{:});
20 end
21
22 function helpWindow_OpeningFcn(hObject, eventdata, handles, varargin)
23 %Executes just before helpTest is made visible.
24
25 %helpID
26 helpID=varargin{1};
27
28 %Pick the corresponding text to be displayed on the window
29 textToBeWritten=helpText(helpID);
30
31 %Display text
32 set(handles.text1,'string',textToBeWritten);
33
34 % Choose default command line output for helpTest
35 handles.output = hObject;
36
37 % Update handles structure
38 guidata(hObject, handles);
39
```

```
% UIWAIT makes helpTest wait for user response (see UIRESUME)
 uiwait(handles.figure1);

function varargout = helpWindow_OutputFcn(hObject, eventdata, handles)
%Outputs from this function are returned to the command line.

function pushbutton1_Callback(hObject, eventdata, handles)
%Executes on button press in pushbutton1.
delete(handles.figure1);

function figure1_CloseRequestFcn(hObject, eventdata, handles)
%Executes when user attempts to close figure1.delete(hObject);
delete(hObject)
```

#### *D.1.2.6 architectureDefinition.m*

```
function varargout = architectureDefinition(varargin)
%Display initial morphological matrix and allows users to define variables
%for each feature

%Begin initialization code
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
    'gui_Singleton',  gui_Singleton, ...
    'gui_OpeningFcn', @architectureDefinition_OpeningFcn, ...
    'gui_OutputFcn',  @architectureDefinition_OutputFcn, ...
    'gui_LayoutFcn',  [] , ...
    'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end

function architectureDefinition_OpeningFcn(hObject, eventdata, ...
    handles, varargin)
%Executes just before architectureDefinition is made visible.

%Display picture
imshow('logo.png');

%Load inputs
```

```
handles.Morpho=varargin{1};
handles.rowNames=varargin{2};
handles.columnNames=varargin{3};
handles.listOptions=varargin{4};
handles.CompatMat=varargin{5};

%Assign inputs to their corresponding tables
set(handles.uitable1,'data',handles.Morpho);
set(handles.uitable1,'rowname',handles.rowNames);
set(handles.uitable1,'columnname',handles.columnNames);

%Define vector that represents the number of options of each feature
handles.numberOptions=[];
for i=1:size(handles.Morpho,1)
    counter=0;
    for j=1:size(handles.Morpho,2)
        if strcmp(handles.Morpho{i,j},'')

        else
            counter=counter+1;
        end
    end
    handles.numberOptions(i)= counter;
end

%Create a big matrix for variables/description
numbVariables=200;
d=length(handles.rowNames);
C=cell(numbVariables,d);
C(:)={''};
set(handles.uitable4, 'data', C);
set(handles.uitable5, 'data', C);

%Create a big matrix for variables assignment
set(handles.uitable6,'ColumnFormat',{'logical'},'ColumnEditable', true);
listOptions=GenerateCompatibility(get(handles.uitable1,'data'),...
    get(handles.uitable1,'data'));
dataVariables=cell(length(listOptions),numbVariables);
dataVariables(:)={false};
set(handles.uitable6,'data',dataVariables);
set(handles.uitable6,'rowname',listOptions);

%Create popup menu
set(handles.popupmenu1,'String', handles.rowNames);

%Choose default command line output for architectureDefinition
handles.output = hObject;

%Update handles structure
```

```
guidata(hObject, handles);

function varargout = architectureDefinition_OutputFcn(hObject, eventdata,...
    handles)
%Outputs from this function are returned to the command line.
%Get default command line output from handles structure
varargout{1} = handles.output;

function popupmenu1_Callback(hObject, eventdata, handles)
%Executes on selection change in popup menu
%Identify selected feature
val = get(hObject,'Value');
listOptions=GenerateCompatibility(get(handles.uitable1,'data'),...
    get(handles.uitable1,'data'));

%Update tables
set(handles.uitable6,'rowname',listOptions);
outputtt=defineFeatures(handles.uitable1, handles.uitable4,...
    handles.uitable5, val, handles.uitable6,handles.numberOptions);
set(handles.uitable4,'data',outputtt{1});
set(handles.uitable5,'data',outputtt{2});
set(handles.uitable6,'data',outputtt{3});
guidata(hObject,handles);

function popupmenu1_CreateFcn(hObject, eventdata, handles)
%Executes during creation of popup menu
if ispc && isequal(get(hObject,'BackgroundColor'),...
        get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function pushbutton1_Callback(hObject, eventdata, handles)
%Executes on button press in "Generate Architectures"
ArchitecturesDisplay(handles.Morpho, get(handles.CompatMat,'data'),...
    handles.figure1,handles.rowNames,get(handles.uitable4,'data'),...
    get(handles.uitable6,'data'),get(handles.uitable5,'data'));
```

#### *D.1.2.7 defineFeatures.m*

```
function varargout = defineFeatures(varargin)
%Display feature name and allows users to add variables, descriptions and
%assign variables to the corresponding features

% Begin initialization code
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
```

```
8       'gui_Singleton',  gui_Singleton, ...
9       'gui_OpeningFcn', @defineFeatures_OpeningFcn, ...
10      'gui_OutputFcn',  @defineFeatures_OutputFcn, ...
11      'gui_LayoutFcn',  [] , ...
12      'gui_Callback',   []);
13  if nargin && ischar(varargin{1})
14      gui_State.gui_Callback = str2func(varargin{1});
15  end
16
17  if nargout
18      [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
19  else
20      gui_mainfcn(gui_State, varargin{:});
21  end
22
23  function defineFeatures_OpeningFcn(hObject, eventdata, handles, varargin)
24  %Executes just before defineFeatures is made visible.
25
26  %Display picture
27  imshow('logo.png')
28
29  %Load inputs
30  handles.morphoInitial=varargin{1};
31  handles.variableTable=varargin{2};
32  handles.descriptionTable=varargin{3};
33  handles.currentFeature=varargin{4};
34  handles.variableAssignment=varargin{5};
35  handles.numberOptions=varargin{6};
36  handles.listFeatures=get(handles.morphoInitial,'rowname');
37
38  %Set current strings to text editors
39  textToBeWritten='Name: ';
40  textToBeWritten2=handles.listFeatures(handles.currentFeature);
41  set(handles.text2,'String',cat(2,textToBeWritten,textToBeWritten2{1}));
42  set(handles.edit2,'String','');
43  set(handles.edit3,'String','');
44
45  %Build variable description table based on current variables
46  variables=get(handles.variableTable,'data');
47  description=get(handles.descriptionTable,'data');
48  combination=[variables(:,handles.currentFeature) ...
49      description(:,handles.currentFeature)];
50  set(handles.uitable1,'data',combination);
51
52  %Build variable assignement table based on current data
53  handles.listOptions=get(handles.morphoInitial,'data');
54  listOptions=handles.listOptions(handles.currentFeature,:);
55  listOptionsWithoutEmpty=listOptions;
56  emptyCells = cellfun(@isempty,listOptionsWithoutEmpty);
```

```
57 listOptionsWithoutEmpty(emptyCells) = [];
58 listOptions=listOptionsWithoutEmpty;
59 set(handles.uitable3,'columnname',variables(:,handles.currentFeature));
60 set(handles.uitable3,'rowname',listOptions);
61
62 jCol=0;
63 while 1-(strcmp(variables{jCol+1,handles.currentFeature},''))
64     jCol=jCol+1;
65 end
66 iRow=length(listOptions);
67 data=cell(iRow,jCol);
68
69 %Load uitable6 with variable assignment
70 if jCol==0
71     set(handles.uitable3,'visible','off');
72     set(handles.uitable3,'data', data,'ColumnFormat',{'logical'},...
73         'ColumnEditable', true);
74 else
75     set(handles.uitable3,'visible','on');
76     entireVariableAssignment=get(handles.variableAssignment,'data');
77     Idx=1;
78     finalIdx=0;
79     if handles.currentFeature==1
80         Idx=1;
81         finalIdx=Idx+handles.numberOptions(1)-1;
82     else
83         for i=1:handles.currentFeature-1
84             Idx=Idx+handles.numberOptions(i);
85             finalIdx=Idx+handles.numberOptions(i+1)-1;
86         end
87     end
88
89     for k=Idx:finalIdx
90         for m=1:jCol
91             tempp=entireVariableAssignment(k,m);
92             data{k-Idx+1,m}=tempp{1};
93         end
94     end
95     set(handles.uitable3,'data', data,'ColumnFormat',{'logical'},...
96         'ColumnEditable', true,'visible','on');
97 end
98
99 % Choose default command line output for defineFeatures
100 handles.output = hObject;
101
102 % Update handles structure
103 guidata(hObject, handles);
104
105 % UIWAIT makes defineFeatures wait for user response
```

```
106 uiwait(handles.figure1);
107
108 function varargout = defineFeatures_OutputFcn(hObject, eventdata, handles)
109 %Outputs from this function are returned to the command line.
110
111 %Load data that are currently in variable table, description tables and
112 %assignment table
113 out1=get(handles.variableTable,'data');
114 out2=get(handles.descriptionTable, 'data');
115 dataAss=get(handles.variableAssignment,'data');
116 temp=get(handles.uitable3,'data');
117
118 %Create the output structure: 1. list of variables, 2. description, 3.
119 %assignment
120 Idx=1;
121 if handles.currentFeature==1
122     Idx=1;
123 else
124     for i=1:handles.currentFeature-1
125         Idx=Idx+handles.numberOptions(i);
126     end
127 end
128
129 for i=1:size(temp,1)
130     for j=1:size(temp,2)
131         dataAss(i+Idx-1,j)=temp(i,j);
132     end
133 end
134 set(handles.variableAssignment,'data',dataAss);
135 out3=get(handles.variableAssignment, 'data');
136 handles.output={out1,out2,out3};
137 varargout{1} = handles.output;
138 delete(handles.figure1);
139
140 function edit2_Callback(hObject, eventdata, handles)
141 %Object declaration for variable name
142
143 function edit2_CreateFcn(hObject, eventdata, handles)
144 %Executes during object creation of edit2 (variable name)
145 if ispc && isequal(get(hObject,'BackgroundColor'),...
146         get(0,'defaultUicontrolBackgroundColor'))
147     set(hObject,'BackgroundColor','white');
148 end
149
150 function edit3_Callback(hObject, eventdata, handles)
151 %Object declaration for description
152
153 function edit3_CreateFcn(hObject, eventdata, handles)
154 %Executes during object creation of edit3 (description)
```

```
155 if ispc && isequal(get(hObject,'BackgroundColor'),...
156         get(0,'defaultUicontrolBackgroundColor'))
157     set(hObject,'BackgroundColor','white');
158 end
159
160 function edit4_Callback(hObject, eventdata, handles)
161 %Object declaration for edit4
162
163 function edit4_CreateFcn(hObject, eventdata, handles)
164 %Executes during object creation of edit4
165 if ispc && isequal(get(hObject,'BackgroundColor'),...
166         get(0,'defaultUicontrolBackgroundColor'))
167     set(hObject,'BackgroundColor','white');
168 end
169
170 function figure1_CloseRequestFcn(hObject, eventdata, handles)
171 %Executes when user attempts to close figure1.
172 guidata(hObject,handles);
173 uiresume();
174
175 function pushbutton1_Callback(hObject, eventdata, handles)
176 %Executes on button press in "Submit".
177
178 %Change variables and description
179 content=get(handles.uitable1,'data');
180 newVariables=content(:,1);
181 newDdescriptions=content(:,2);
182
183 variables=get(handles.variableTable,'data');
184 descriptions=get(handles.descriptionTable,'data');
185
186 for i=1:size(newVariables,1)
187     variables(i,handles.currentFeature)=newVariables(i);
188     descriptions(i,handles.currentFeature)=newDdescriptions(i);
189 end
190
191 set(handles.variableTable,'data',variables);
192 set(handles.descriptionTable,'data',descriptions);
193 set(handles.uitable3,'columnname',variables);
194
195 guidata(hObject,handles);
196 uiresume();
197
198 function pushbutton2_Callback(hObject, eventdata, handles)
199 %Executes on button press in "Add variable".
200
201 %Load text in variable characteristics
202 handles.varNames=get(handles.edit2,'String');
203 handles.varDes=get(handles.edit3,'String');
```

```
204
205 if 1-strcmp(handles.varNames,'') && 1-strcmp(handles.varNames,' ') &&...
206         1-strcmp(handles.varNames,'  ') ...
207         && 1-strcmp(handles.varNames,'   ') &&...
208         1-strcmp(handles.varNames,'    ') &&...
209         1-strcmp(handles.varNames,'     ') &&...
210         1-strcmp(handles.varNames,'      ')
211
212     %Store characteristics in uitable
213     data=get(handles.uitable1,'data');
214     %newRow={handles.varNames,handles.varDes};
215     Idx=1;
216     allVar=[];
217     while strcmp(data{Idx,1},'')==0 && strcmp(data{Idx,1},'')==0
218         allVar{Idx}=data{Idx,1};
219         Idx=Idx+1;
220     end
221     %WhereToStore=data;
222     data{Idx,1}=handles.varNames;
223     data{Idx,2}=handles.varDes;
224     set(handles.uitable1,'data',data);
225     allVar{Idx}=data{Idx,1};
226
227     dataTemp=get(handles.uitable3,'data');
228     addCol=cell(size(dataTemp,1),1);
229     addCol(:)={false};
230     dataTemp=[dataTemp addCol];
231
232     set(handles.uitable3,'visible','on');
233     set(handles.uitable3,'columnname',allVar);
234     set(handles.uitable3,'data',dataTemp);
235 end
236
237 %Empty text editors
238 set(handles.edit2,'String','');
239 set(handles.edit3,'String','');
240
241 %Update guidata
242 guidata(hObject, handles);
243
244 function uipushtool1_ClickedCallback(hObject, eventdata, handles)
245 %Load table with variables
246 warning('off','MATLAB:table:ModifiedVarnames');
247 possiblefile={'*.xlsx';'*.xls'};
248 [FileName,PathName] = uigetfile(possiblefile,'Select an Excel file');
249
250 if FileName == 0
251     panda = 0;
252 else
```

```
253     filePathName=cat(2,PathName,FileName);
254     T=readtable(filePathName);
255     set(handles.uitable1,'data',table2cell(T));
256     handles.rowNames= T{:,1};
257 end
258 
259 %Update assignment matrix
260 handles.listOptions=get(handles.morphoInitial,'data');
261 listOptions=handles.listOptions(handles.currentFeature,:);
262 listOptionsWithoutEmpty=listOptions;
263 emptyCells = cellfun(@isempty,listOptionsWithoutEmpty);
264 listOptionsWithoutEmpty(emptyCells) = [];
265 listOptions=listOptionsWithoutEmpty;
266 variables=get(handles.uitable1,'data');
267 set(handles.uitable3,'columnname',variables(:,1));
268 set(handles.uitable3,'rowname',listOptions);
269 jCol=size(variables,1);
270 iRow=length(listOptions);
271 data=cell(iRow,jCol);
272 data(:)={false};
273 
274 %Load existing assignment table with variable assignment
275 if jCol==0
276     set(handles.uitable3,'visible','off');
277     set(handles.uitable3,'data', data,'ColumnFormat',...
278         {'logical'},'ColumnEditable', true);
279 else
280     set(handles.uitable3,'visible','on');
281     set(handles.uitable3,'data', data,'ColumnFormat',...
282         {'logical'},'ColumnEditable', true,'visible','on');
283 end
284 
285 guidata(hObject, handles);
286 
287 function uipushtool2_ClickedCallback(hObject, eventdata, handles)
288 %Save table with variables
289 possiblefile={'*.xlsx';'*.xls'};
290 [FileName,PathName] = uiputfile(possiblefile);
291 
292 if FileName == 0
293 
294 else
295     filePathName=cat(2,PathName,FileName);
296     headers=get(handles.uitable1, 'columnname');
297     dataToBeWritten=get(handles.uitable1, 'data');
298     matrixToBeWritten=cat(1,headers',dataToBeWritten);
299     a = dir(PathName);
300     b = struct2cell(a);
301     if any(ismember(b(1,:),FileName)) == 1
```

```
302         EraseExcelSheets(filePathName);
303     end
304     xlswrite(filePathName,matrixToBeWritten);
305 end
306
307 function uipushtool3_ClickedCallback(hObject, eventdata, handles)
308 %Help window
309 helpWindow('assignVariables');
```

#### *D.1.2.8 ArchitecturesDisplay.m*

```
1  function varargout = ArchitecturesDisplay(varargin)
2  %Display feasible architectures
3
4  %Begin initialization code
5  gui_Singleton = 1;
6  gui_State = struct('gui_Name',       mfilename, ...
7      'gui_Singleton',  gui_Singleton, ...
8      'gui_OpeningFcn', @ArchitecturesDisplay_OpeningFcn, ...
9      'gui_OutputFcn',  @ArchitecturesDisplay_OutputFcn, ...
10     'gui_LayoutFcn',  [] , ...
11     'gui_Callback',   []);
12 if nargin && ischar(varargin{1})
13     gui_State.gui_Callback = str2func(varargin{1});
14 end
15
16 if nargout
17     [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
18 else
19     gui_mainfcn(gui_State, varargin{:});
20 end
21
22 function ArchitecturesDisplay_OpeningFcn(hObject, eventdata, handles,...
23     varargin)
24 %Executes just before ArchitecturesDisplay is made visible
25
26 %Display picture
27 imshow('logo.png')
28
29 %Load inputs
30 MorphoTest=varargin{1};
31 CompatMatrix1=varargin{2};
32 handles.rowNames=varargin{4};
33 handles.variableMat=varargin{5};
34 handles.varAss=varargin{6};
35 handles.varDes=varargin{7};
```

```
d=size(CompatMatrix1);
for i=1:d(1)
    for j=1:d(2)
        a=cell2mat(CompatMatrix1(i,j));
        CompatMatrix{i,j}=num2str(a);
    end
end

%Generate feasible architectures
tempp=GenerateFeasibleArchitectures(MorphoTest,CompatMatrix,...
    handles.varAss,handles.variableMat);
possAlt=tempp{2};
feasAlt=tempp{3};
temp=tempp{1};

%Display feasible architectures
handles.ListFeasibleAlternatives=temp{1};
set(handles.uitable1,'data', handles.ListFeasibleAlternatives);
set(handles.uitable1, 'columnname',handles.rowNames);
set(handles.uitable1,'rowname',temp{3});
handles.finalRow=temp{3};

%Write total number of alternatives and architectures
textToBeWritten='Number of feasible architectures: ';
textToBeWritten=cat(2,textToBeWritten,num2str(temp{2}));
set(handles.text2, 'String', textToBeWritten);
textToBeWritten1='Number of feasible alternatives: ';
textToBeWritten2='Number of possible alternatives: ';
textToBeWrittenTop=cat(2,textToBeWritten1,num2str(feasAlt));
textToBeWrittenDown=cat(2,textToBeWritten2,num2str(possAlt));
set(handles.text3, 'String', textToBeWrittenTop);
set(handles.text4, 'String', textToBeWrittenDown);

% Choose default command line output for ArchitecturesDisplay
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);

function varargout = ArchitecturesDisplay_OutputFcn(hObject, eventdata,...
    handles)
%Outputs from this function are returned to the command line.
%Get default command line output from handles structure
varargout{1} = handles.output;

function uipushtool1_ClickedCallback(hObject, eventdata, handles)
%Save data table
possiblefile={'*.xlsx';'*.xls'};
[FileName,PathName] = uiputfile(possiblefile);
```

```
85
86 if FileName == 0
87
88 else
89     filePathName=cat(2,PathName,FileName);
90     headers=get(handles.uitable1, 'columnname');
91     dataToBeWritten=get(handles.uitable1, 'data');
92     rowHeader=[' ';handles.finalRow'];
93     matrixToBeWritten=cat(1,headers',dataToBeWritten);
94     matrixToBeWritten=cat(2,rowHeader,matrixToBeWritten);
95     a = dir(PathName);
96     b = struct2cell(a);
97     if any(ismember(b(1,:),FileName)) == 1
98         EraseExcelSheets(filePathName);
99     end
100    xlswrite(filePathName,matrixToBeWritten);
101 end
102
103 function pushbutton1_Callback(hObject, eventdata, handles)
104 %Executes on button press in "Variable description".
105 DisplayDescription(handles.variableMat,handles.varDes);
```

#### *D.1.2.9 changeFeatureName.m*

```
1  function varargout = changeFeatureName(varargin)
2  %Change the name of a feature
3
4  %Begin initialization code
5  gui_Singleton = 1;
6  gui_State = struct('gui_Name',       mfilename, ...
7      'gui_Singleton',  gui_Singleton, ...
8      'gui_OpeningFcn', @changeFeatureName_OpeningFcn, ...
9      'gui_OutputFcn',  @changeFeatureName_OutputFcn, ...
10     'gui_LayoutFcn',  [] , ...
11     'gui_Callback',   []);
12 if nargin && ischar(varargin{1})
13     gui_State.gui_Callback = str2func(varargin{1});
14 end
15
16 if nargout
17     [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
18 else
19     gui_mainfcn(gui_State, varargin{:});
20 end
21
22 function changeFeatureName_OpeningFcn(hObject,eventdata,handles,varargin)
```

```
%Executes just before changeFeatureName is made visible.
%Load current feature name
handles.featureName=varargin{1};
set(handles.edit1,'String',handles.featureName);

%Choose default command line output for changeFeatureName
handles.output = hObject;

%Update handles structure
guidata(hObject, handles);

%UIWAIT makes changeFeatureName wait for user response
uiwait(handles.figure1);

function varargout = changeFeatureName_OutputFcn(hObject,eventdata,handles)
%Outputs from this function are returned to the command line.
%Get default command line output from handles structure
varargout{1} = handles.output;
delete(handles.figure1);

function edit1_Callback(hObject, eventdata, handles)
%Declaration of the field for text edition

function edit1_CreateFcn(hObject, eventdata, handles)
%Executes during edit1 creation
if ispc && isequal(get(hObject,'BackgroundColor'),...
        get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function pushbutton1_Callback(hObject, eventdata, handles)
%Executes on button press in "Ok"
%Load text from edit1
handles.featureName=get(handles.edit1,'string');

%Output text loaded
handles.output=handles.featureName;
guidata(hObject,handles);
uiresume();

function figure1_CloseRequestFcn(hObject, eventdata, handles)
%Executes when user attempts to close figure1.
handles.output=handles.featureName;
guidata(hObject,handles);
uiresume();
```

#### *D.1.2.10 noMorpho.m*

```
function varargout = noMorpho(varargin)
%Error message when the morphological matrix is empty

%Begin initialization code
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @noMorpho_OpeningFcn, ...
                   'gui_OutputFcn',  @noMorpho_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end

function noMorpho_OpeningFcn(hObject, eventdata, handles, varargin)
%Executes just before noMorpho is made visible.

% Choose default command line output for noMorpho
handles.output = 'Yes';

% Update handles structure
guidata(hObject, handles);

% Insert custom Title and Text
if(nargin > 3)
    for index = 1:2:(nargin-3),
        if nargin-3==index, break, end
        switch lower(varargin{index})
         case 'title'
          set(hObject, 'Name', varargin{index+1});
         case 'string'
          set(handles.text1, 'String', varargin{index+1});
        end
    end
end

% Determine the position of the dialog - centered on the callback figure
% if available, else, centered on the screen
FigPos=get(0,'DefaultFigurePosition');
OldUnits = get(hObject, 'Units');
set(hObject, 'Units', 'pixels');
OldPos = get(hObject,'Position');
```

```
FigWidth = OldPos(3);
FigHeight = OldPos(4);
if isempty(gcbf)
    ScreenUnits=get(0,'Units');
    set(0,'Units','pixels');
    ScreenSize=get(0,'ScreenSize');
    set(0,'Units',ScreenUnits);

    FigPos(1)=1/2*(ScreenSize(3)-FigWidth);
    FigPos(2)=2/3*(ScreenSize(4)-FigHeight);
else
    GCBFOldUnits = get(gcbf,'Units');
    set(gcbf,'Units','pixels');
    GCBFPos = get(gcbf,'Position');
    set(gcbf,'Units',GCBFOldUnits);
    FigPos(1:2) = [(GCBFPos(1) + GCBFPos(3) / 2) - FigWidth / 2, ...
                   (GCBFPos(2) + GCBFPos(4) / 2) - FigHeight / 2];
end
FigPos(3:4)=[FigWidth FigHeight];
set(hObject, 'Position', FigPos);
set(hObject, 'Units', OldUnits);


% Make the GUI modal
set(handles.figure1,'WindowStyle','modal')

% UIWAIT makes noMorpho wait for user response (see UIRESUME)
uiwait(handles.figure1);

function varargout = noMorpho_OutputFcn(hObject, eventdata, handles)
%Outputs from this function are returned to the command line.

function pushbutton1_Callback(hObject, eventdata, handles)
%Executes on button press in pushbutton1.
delete(hObject);

function figure1_CloseRequestFcn(hObject, eventdata, handles)
%Executes when user attempts to close figure1.
delete(hObject);

function pushbutton3_Callback(hObject, eventdata, handles)
%Executes on button press in "Ok"
delete(handles.figure1);
```

#### *D.1.2.11 wrongCompat.m*

```
function varargout = wrongCompat(varargin)
%Display error message when an inconsistent compatibility matrix is
%provided by the user

%Begin initialization code
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @wrongCompat_OpeningFcn, ...
                   'gui_OutputFcn',  @wrongCompat_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end

function wrongCompat_OpeningFcn(hObject, eventdata, handles, varargin)
%Executes just before wrongCompat is made visible.

%Choose default command line output for wrongCompat
handles.output = 'Yes';

%Update handles structure
guidata(hObject, handles);

%Insert custom Title and Text
if(nargin > 3)
    for index = 1:2:(nargin-3),
        if nargin-3==index, break, end
        switch lower(varargin{index})
         case 'title'
          set(hObject, 'Name', varargin{index+1});
         case 'string'
          set(handles.text1, 'String', varargin{index+1});
        end
    end
end

%Determine the position of the dialog - centered on the callback figure
%if available, else, centered on the screen
FigPos=get(0,'DefaultFigurePosition');
OldUnits = get(hObject, 'Units');
set(hObject, 'Units', 'pixels');
```

```
OldPos = get(hObject,'Position');
FigWidth = OldPos(3);
FigHeight = OldPos(4);
if isempty(gcbf)
    ScreenUnits=get(0,'Units');
    set(0,'Units','pixels');
    ScreenSize=get(0,'ScreenSize');
    set(0,'Units',ScreenUnits);

    FigPos(1)=1/2*(ScreenSize(3)-FigWidth);
    FigPos(2)=2/3*(ScreenSize(4)-FigHeight);
else
    GCBFOldUnits = get(gcbf,'Units');
    set(gcbf,'Units','pixels');
    GCBFPos = get(gcbf,'Position');
    set(gcbf,'Units',GCBFOldUnits);
    FigPos(1:2) = [(GCBFPos(1) + GCBFPos(3) / 2) - FigWidth / 2, ...
                   (GCBFPos(2) + GCBFPos(4) / 2) - FigHeight / 2];
end
FigPos(3:4)=[FigWidth FigHeight];
set(hObject, 'Position', FigPos);
set(hObject, 'Units', OldUnits);

%Make the GUI modal
set(handles.figure1,'WindowStyle','modal')

%UIWAIT makes wrongCompat wait for user response
uiwait(handles.figure1);

function varargout = wrongCompat_OutputFcn(hObject, eventdata, handles)
%Outputs from this function are returned to the command line.

function pushbutton1_Callback(hObject, eventdata, handles)
%Executes on button press in pushbutton1.

handles.output = get(hObject,'String');

%Update handles structure
guidata(hObject, handles);

%Use UIRESUME instead of delete because the OutputFcn needs
%to get the updated handles structure.
uiresume(handles.figure1);

function figure1_CloseRequestFcn(hObject, eventdata, handles)
%Executes when user attempts to close figure1.
delete(hObject);

function pushbutton3_Callback(hObject, eventdata, handles)
```

```
%Executes on button press in pushbutton3.
delete(handles.figure1);
```

#### *D.1.2.12 DisplayDescription.m*

```
function varargout = DisplayDescription(varargin)
%Display the description of all variables

%Begin initialization code
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
    'gui_Singleton',  gui_Singleton, ...
    'gui_OpeningFcn', @DisplayDescription_OpeningFcn, ...
    'gui_OutputFcn',  @DisplayDescription_OutputFcn, ...
    'gui_LayoutFcn',  [] , ...
    'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end

function DisplayDescription_OpeningFcn(hObject, eventdata, handles,...
    varargin)
%Executes just before DisplayDescription is made visible.

%Load data data table
variablesTable=varargin{1};
descriptionTable=varargin{2};
tableToBeDisplayed=generateVariableDescription(variablesTable,...
    descriptionTable);
set(handles.uitable1,'data',tableToBeDisplayed);


%Choose default command line output for DisplayDescription
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);

function varargout = DisplayDescription_OutputFcn(hObject, eventdata,...
    handles)
```

```
%Outputs from this function are returned to the command line.

function pushbutton1_Callback(hObject, eventdata, handles)
%Executes on button press "Ok"
delete(handles.figure1);

function figure1_CloseRequestFcn(hObject, eventdata, handles)
%Executes when user attempts to close figure1.
delete(hObject);

function uipushtool1_ClickedCallback(hObject, eventdata, handles)
%Save table with variables
possiblefile={'*.xlsx';'*.xls'};
[FileName,PathName] = uiputfile(possiblefile);

if FileName == 0

else
    filePathName=cat(2,PathName,FileName);
    headers={'Variables' 'Description'};
    dataToBeWritten=get(handles.uitable1, 'data');
    matrixToBeWritten=cat(1,headers,dataToBeWritten);
    a = dir(PathName);
    b = struct2cell(a);
    if any(ismember(b(1,:),FileName)) == 1
        EraseExcelSheets(filePathName);
    end
    xlswrite(filePathName,matrixToBeWritten);
end
```

#### *D.1.2.13 helpText.m*

```
function helpText = helpText(helpID)
%Define texts that are loaded to the help window depending on the helpID
%that is called
switch helpID
    case 'morpho'
        helpText={
            'A morphological matrix is a systematic and rigorous method';
            'for generating all possible alternatives of a given system.';
            ' ';
            'The system must first be decomposed into features';
            '(or functions). You can edit the feature names using the';
            'popup menu in the top right-hand corner of the window.';
            ' ';
            'Once, all features have been added to the matrix, start';
```

```
15              'adding options for each feature. These options correspond to';
16              'the different ways the feature (function) can be fulfilled.';
17              ' ';
18              'Using the toolbar, you can easily load an existing';
19              'morphological matrix from an Excel sheet and save your own';
20              'matrix in an Excel file.';
21              ' ';
22              'Once the morphological matrix has been completed, press';
23              'Ensure compatibility to define the compatibility between';
24              'each option of the system.'};
25      case 'compat'
26          helpText={
27              'A compatibility matrix is a systematic and rigorous method';
28              'for defining the compatibility between all options of';
29              'a system.';
30              ' ';
31              'If the options k and m are compatible, write  1  in the';
32              'cell (k,m). If these options are not compatible, write  0 .';
33              'The compatibility matrix is symmetric so that you only need';
34              'to complete the upper triangular (note that the lower';
35              'triangular will not be considered in the calculation and';
36              'that all values other than  1  will be converted to  0  for';
37              'the calculation. The compatibility matrix has been';
38              'automatically prefilled in order to account for the';
39              'incompatibility between the options of the same feature.';
40              ' ';
41              'Using the toolbar, you can easily load an existing';
42              'compatibility matrix from an Excel sheet and save your own';
43              'matrix in an Excel file.';
44              ' ';
45              'Once the compatibility matrix is completed, press either';
46              'Generate alternatives to generate the list of feasible';
47              'alternatives or Generate architectures to define design';
48              'variables and generate the list of feasible architectures.'};
49      case 'assignVariables'
50          helpText={
51              'This window allows you to define all design variables';
52              'corresponding to the selected feature along with their';
53              'description.';
54              ' ';
55              'Once all design variables have been added to the list,'
56              'assign variables to the corresponding options using the';
57              'checkboxes. Using the toolbar, you can easily load an';
58              'existing list of variables from an Excel sheet and save your';
59              'own list in an Excel file.';
60              ' ';
61              'Once all variables have been added and assigned, press';
62              'Submit  to save your modification and go back to the';
63              'previous window.'};
```

```
end

end
```

#### *D.1.2.14 PreFillCompa.m*

```
function compatMat = PreFillCompa(compatMatEmpty,Morpho )
%Prefill the morphological matrix with 1 on the diagonal and 0 for options
%from the same feature

%Add 1 on diagonal
d=size(compatMatEmpty);
for i=1:d(1)
    compatMatEmpty{i,i}='1';
end

%Put 0 for options from the same feature
dM=size(Morpho);
VectorAlternatives=[];
for i=1:dM(1)
    compatRowsWithoutEmpty = Morpho(i,:);
    emptyCells = cellfun(@isempty,compatRowsWithoutEmpty);
    compatRowsWithoutEmpty(emptyCells) = [];
    VectorAlternatives(i)=length(compatRowsWithoutEmpty);
end
counter2=0;
for i=1:length(VectorAlternatives)
    for j=1:VectorAlternatives(i)
        for k=1:VectorAlternatives(i)-j
            compatMatEmpty{j+counter2,j+counter2+k}='0';
        end
    end
    counter2=counter2+VectorAlternatives(i);
end

compatMat=compatMatEmpty;
end
```

#### *D.1.2.15 GenerateCompatibility.m*

```
function listOptions  = GenerateCompatibility( MorphoTest, MorphoVar )
%Create a list of options from a matrix

```

```
d=size(MorphoTest);
compatRows=[];
compatRows2=[];
for i=1:d(1)
    compatRows=cat(2,compatRows,MorphoTest(i,:));
    compatRows2=cat(2,compatRows2,MorphoVar(i,:));
end

%Remove empty cells
compatRowsWithoutEmpty=compatRows;
compatRowsWithoutEmpty2=compatRows2;
emptyCells = cellfun(@isempty,compatRowsWithoutEmpty);
compatRowsWithoutEmpty2(emptyCells) = [];
listOptions=compatRowsWithoutEmpty2;
end
```

#### *D.1.2.16 GenerateFeasibleAlternatives.m*

```
function output = GenerateFeasibleAlternatives( MorphoTest,compatMat )
%Generate the list of feasible alternatives

%Create the list of possible options based on the morphological matrix
listOptions=GenerateCompatibility( MorphoTest,MorphoTest );

global endSet
endSet=[];

%Set of compatible options
numberOptions=[];
for i=1:size(MorphoTest,1)
    counter=0;
    for j=1:size(MorphoTest,2)
        if strcmp(MorphoTest{i,j},'')

        else
            counter=counter+1;
        end
    end
    numberOptions(i)= counter;
end

%Waiting bar
h = waitbar(0,'Please wait...');
steps=numberOptions(1);
for fLine=1:numberOptions(1)
    waitbar(fLine / steps)
```

```
29      AlgoCompa( compatMat,MorphoTest, 2,[fLine],numberOptions);
30  end
31  close(h)
32
33  %Create the matrix of strings to be displayed in GUI from endSet
34  finalSetOfCombinations=[];
35
36  for i=1:size(endSet,1)
37      for j=1:size(endSet,2)
38          temp=endSet(i,j);
39          finalSetOfCombinations{i,j}=listOptions{temp};
40      end
41  end
42
43  %Generate outputs
44  out1=finalSetOfCombinations;
45  out2=prod(numberOptions);
46  output={out1,out2};
47  end
```

#### *D.1.2.17 GenerateFeasibleArchitectures.m*

```
1   function output = GenerateFeasibleArchitectures( MorphoTest,...
2       compatMat,varAss,variableMat )
3
4   %Initialization of list of options from morphological matrix
5   listOptions=GenerateCompatibility( MorphoTest,MorphoTest );
6
7   global endSet
8   endSet=[];
9
10  %Set of compatible options
11  numberOptions=[];
12  for i=1:size(MorphoTest,1)
13      counter=0;
14      for j=1:size(MorphoTest,2)
15          if strcmp(MorphoTest{i,j},'')
16
17          else
18              counter=counter+1;
19          end
20      end
21      numberOptions(i)= counter;
22  end
23
24  %Generate morphological matrix of variables
```

```
MorphoVar=GenerateMorphoVar(varAss,numberOptions,variableMat);
MorphoVarVar=GenerateMorphoVar2(varAss,numberOptions,variableMat);


%If no variable has been selected for a given feature, put ' '
emptyIndex = cellfun(@isempty,MorphoVar);    %# Find indices of empty cells
MorphoVar(emptyIndex) = {' '};                 %# Fill empty cells with ' '
emptyIndex = cellfun(@isempty,MorphoVarVar);
MorphoVarVar(emptyIndex) = {' '};

listOptions2=GenerateCompatibility( MorphoTest, MorphoVar );
listOptionsVar=GenerateCompatibility( MorphoTest, MorphoVarVar );

%Waiting bar
h = waitbar(0,'Please wait...');
steps=numberOptions(1);
for fLine=1:numberOptions(1)
    waitbar(fLine / steps)
    AlgoCompa( compatMat,MorphoTest, 2,[fLine],numberOptions);
end
close(h)
%Create the matrix of strings to be displayed in GUI from endSet
finalSetOfCombinations=[];
finalSetOfCombinations2=[];
finalSetOfCombinationsVar=[];
for i=1:size(endSet,1)
    for j=1:size(endSet,2)
        temp=endSet(i,j);
        finalSetOfCombinations{i,j}=listOptions{temp};
        finalSetOfCombinations2{i,j}=listOptions2{temp};
        finalSetOfCombinationsVar{i,j}=listOptionsVar{temp};
    end
end

%Create the table with architecture
variableFeasible= detectArchitectures( finalSetOfCombinations,...
    finalSetOfCombinations2,finalSetOfCombinationsVar );
out1=variableFeasible;
out2=prod(numberOptions);
out3=size(finalSetOfCombinations,1);
output={out1, out2, out3};
end
```

#### *D.1.2.18 generateVariableDescription.m*

```
function outputTable  = generateVariableDescription( variablesTable,...
```

```
    descriptionTable )
%Generate the table of variables and the corresponding descriptions

%Load data and initialize table
d=size(variablesTable);
listVar=[];
listDes=[];

%Create a list based on a matrix
for i=1:d(2)
listVar=cat(2,listVar,variablesTable(:,i));
listDes=cat(2,listDes,descriptionTable(:,i));
end

%Remove empty cells from the list
listVarWithoutEmpty=listVar;
listDesWithoutEmpty=listDes;
emptyCells = cellfun(@isempty,listVarWithoutEmpty);
listVarWithoutEmpty(emptyCells) = [];
listDesWithoutEmpty(emptyCells) = [];

%Save the final lists
listVariables=listVarWithoutEmpty;
listDescription=listDesWithoutEmpty;

%Create the table
outputTable=cell(2,length(listVariables));
outputTable(1,:)=listVariables;
outputTable(2,:)=listDescription;
outputTable=outputTable';

end
```

#### *D.1.2.19 AlgoCompa.m*

```
function AlgoCompa( compatMat,MorphoTest, currentLevel, previousSet,...
    numberOptions)
%Recursive function that searches for compatible combinations of options

global endSet

%%Search loop
%Bottom of the morphological matrix reached?
if currentLevel== (size(MorphoTest,1)+1)

    %We reach the bottom of the morphological matrix. Search is stopped and
```

```
    %the combination of compatible options is saved in the final set
    endSet=[endSet ; previousSet];

else
    %1st element in the compatibility matrix to be investigated
    startCompat=1;
    for k=1:currentLevel-1
        startCompat=startCompat+numberOptions(k);
    end

    %Check compatibility between new element and elements from the previous
    %set
    for j=1:numberOptions(currentLevel)
        if checkCompat( previousSet,compatMat,j,startCompat )==1
            currentSet=cat(2,previousSet,startCompat+j-1);
            AlgoCompa( compatMat,MorphoTest, currentLevel+1, currentSet,...
                numberOptions);
        end
    end

end

end
```

#### *D.1.2.20 detectArchitectures.m*

```
function out = detectArchitectures( finalSetOfCombinations,...
    finalSetOfCombinations2,finalSetOfCombinationsVar )
%Sort the list of feasible alternatives so that alternatives with the same
%design variables (same architecture) are grouped together

%Initialization
toBeCompared=cell(size(finalSetOfCombinations2,1),1);

%Load matrix to be compared
for i=1:length(toBeCompared)
    for j=1:size(finalSetOfCombinations2,2)
        temp=strcat(toBeCompared{i},finalSetOfCombinations2(i,j));
        toBeCompared{i}=temp{1};
    end
end

%Re-order everything
[storageOrdered, order] = sort(toBeCompared);

if isempty(storageOrdered)
```

```
21      storageOrderedFinal=[];
22      finalRows=[];
23      numberArchi=0;
24  else
25      %Reorder alternatives
26      reorderedAlter=cell(size(finalSetOfCombinations,1),...
27          size(finalSetOfCombinations,2));
28
29      %Reorder variables
30      reorderedVar=cell(size(finalSetOfCombinationsVar,1),...
31          size(finalSetOfCombinationsVar,2));
32
33      for i=1:size(finalSetOfCombinations,1)
34          reorderedAlter(i,:)=finalSetOfCombinations(order(i),:);
35          reorderedVar(i,:)=finalSetOfCombinationsVar(order(i),:);
36      end
37
38      %Re-order everything and put architecture number
39      storageOrderedFinal=[];
40      offsetIdx=0;
41
42      %storageOrderedFinal(1,:)=reorderedVar(1,:);
43      finalRows{1}='Architecture 1';
44
45      for j=1:size(reorderedAlter,2)
46          storageOrderedFinal{1,j}=reorderedVar{1,j};
47      end
48
49      for j=1:size(reorderedAlter,2)
50          storageOrderedFinal{2,j}=reorderedAlter{1,j};
51      end
52
53      for i=1:size(storageOrdered,1)-1
54          if 1-strcmp(storageOrdered(i),storageOrdered(i+1))
55              storageOrderedFinal(i+offsetIdx+1,:)=reorderedAlter(i,:);
56              offsetIdx=offsetIdx+1;
57              storageOrderedFinal(i+offsetIdx+1,:)=reorderedVar(i+1,:);
58              finalRows{i+offsetIdx+1}=strcat('Architecture ',...
59                  num2str(offsetIdx+1));
60          else
61              storageOrderedFinal(i+offsetIdx+1,:)=reorderedAlter(i,:) ;
62              finalRows{i+offsetIdx+1}=' ';
63          end
64      end
65      numberArchi=offsetIdx+1;
66      if 1-isequal(size(storageOrdered,1),1)
67          storageOrderedFinal(size(storageOrdered,1)+...
68              offsetIdx+1,:)=reorderedAlter(i+1,:) ;
69          finalRows{size(storageOrdered,1)+offsetIdx+1}=' ';
```

```
    end

end

%Generate output structure
out1=storageOrderedFinal;
out2=numberArchi;
out3=finalRows;
out = {out1, out2,out3};

end
```

#### *D.1.2.21 checkCompat.m*

```
function output = checkCompat( previousSet, compatMat,j,currentLevel )
%Function that checks the compatibility between an option and the ones that
%are stored in the vector

%Initialization
output=1;

%Product of all cells with the new one. Output is 1 if compatible, 0
%otherwise
for i=1:length(previousSet)
    output=output*str2num(compatMat{previousSet(i),currentLevel+j-1});
end

end
```

#### *D.1.2.22 GenerateMorphoVar.m*

```
function  morphoVar = GenerateMorphoVar(varAss,numberOptions,variableMat)
%Generate a matrix similar to the morphological matrix but replace options
%with variable in binary code

%Initialization
morphoVar=[];
Idx=1;

%Loop on the line
for i=1:length(numberOptions); %1st line of the morphological matrix
    nmbVar=1;
    while 1-strcmp(variableMat(nmbVar,i),'')
```

```
13          nmbVar=nmbVar+1;
14      end
15      %loop on the column
16      for j=1:numberOptions(i)
17          temp=[];
18
19          %Fill the matrix
20          for k=1:nmbVar-1
21              tempp=varAss(Idx+j-1,k);
22              temp=cat(2,temp,num2str(tempp{1}));
23          end
24          morphoVar{i,j}=temp;
25      end
26      Idx=Idx+numberOptions(i);
27  end
28  end
```

#### *D.1.2.23 GenerateMorphoVar2.m*

```
1   function  morphoVar2 = GenerateMorphoVar2(varAss,numberOptions,variableMat)
2   %Generate a matrix similar to the morphological matrix but replace options
3   %with variable names
4
5   %Initialization
6   morphoVar2=[];
7   Idx=1;
8
9   %Loop on the line
10  for i=1:length(numberOptions); %1st line of the morphological matrix
11      nmbVar=1;
12      while 1-strcmp(variableMat(nmbVar,i),'')
13          nmbVar=nmbVar+1;
14      end
15
16      %loop on the column
17      for j=1:numberOptions(i)
18          temp=[];
19          virg=0;
20
21          %Fill the matrix
22          for k=1:nmbVar-1
23              tempp=varAss{Idx+j-1,k};
24              if tempp==1
25                  virg=virg+1;
26                  temp=cat(2,temp,variableMat{k,i});
27                  temp=cat(2,temp,', ');
```

```
            end
        end
        if 1-isequal(virg,0)
            temp=temp(1:end-2);
        end
        morphoVar2{i,j}=temp;
    end
    Idx=Idx+numberOptions(i);
end
end
```

## *D.2 Implementation of the Design Framework*

### D.2.1 Weight Module

```
function output= weightModule(Swing, tc, TRwing, ARwing, dfus,...
    wing, nmax, nPilots, nPAX, qmax, tmission, propellantWeight,...
    JetEngine, Tj, nJet, BPR, MmaxJet, afterburner, TIT, fuelWeight,...
    Tr, pc, epsilon, combTime, engineDiameter, Propellant, seatPitch,...
    la, ln, db, verticalTail, horizontalTail)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Weight module: compution of the weight of the vehicle by decomposing it
% into components
% Author: Christopher Frank
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%% Jet propulsion module
if JetEngine == 1
    jetEngine=JetEngines(Tj, BPR, MmaxJet, afterburner, TIT);
    jetEngineWeight=nJet*jetEngine.weight*2.20462; %kg -> lbs
    jetEngineDiameter=jetEngine.diameter;
    jetEngineLength=jetEngine.length;
    jetEngineTSFC=jetEngine.SFC;
else
    jetEngineDiameter=[];
    jetEngineLength=[];
    jetEngineTSFC=[];
    jetEngineWeight=0;
end

%% Rocket propulsion module
mProp=propellantWeight; %Propellant mass (kg)
thrust=Tr; %Convert Tr in Newtons
rocketEngineWeightStruct=RocketEngines(pc, epsilon, mProp, combTime,...
    thrust, engineDiameter, Propellant);
rocketEngineWeight=rocketEngineWeightStruct.weightEngine*2.20462; %kg->lbs
```

```
rocketEngineLength=rocketEngineWeightStruct.lengthEngine;

%% Conversion (from SI to British)
Swing=Swing*10.76391;
qmax=qmax*0.0208854342;
dfus=dfus*3.280840;
la=la*3.280840;
ln=ln*3.280840;
db=db*3.280840;
fuelWeight=fuelWeight*2.204623;
propellantWeight=propellantWeight*2.204623;
Tr=Tr*0.224808943;
b=sqrt(Swing*ARwing);
rootChord=2*Swing/(b*(1+TRwing)); %Wing root chord

%% Coefficients used in the weight estimation model
Kwing=0.214; %Wing material coefficient (assuming composite)
Kct=0.05; %Wing carry-thru constants
eta=0.2; %Wing/body efficiency factor (conventional vehicle)
Kt=1.108; %Tail material coefficient (assuming composite)
Kpc=0.712; %Standard hydraulic system coefficient
Kpe=0.97e-4; %Engine gimbal power demand
massPAX=198; %lbs per PAX
Kpb=0.405; %Battery power demand constant
Khyd=2.1; %Technology base for hydraulic system
Kecd=0.02;
Ksca=3.32; %Coefficient for surface control
cockpitPitch=1.5;
rhoFuel=719; %Fuel density (kg/m^3)

%% Coefficients for near-term improvement provided by Rohrschneider
etaNewTechno=1;
TRFwing=0.44*etaNewTechno;
TRFtail=0.44*etaNewTechno;
TRFbody=0.38*etaNewTechno;
TRFtps=0.35*etaNewTechno;
TRFlg=0.09*etaNewTechno;
TRFavionics=0.5*etaNewTechno;
TRFeclss=0.1*etaNewTechno;

%% Intermediate calculations
Ndays=tmission/(3600*24);
payloadWeight=massPAX*(nPAX+nPilots);
Nz=1.5*nmax;
troot=tc*rootChord;
propulsionWeight=rocketEngineWeight+jetEngineWeight;
fuelPropellantWeight=propellantWeight+fuelWeight;

%% Loop
```

```
%%Initialization
unFuelledMass=1;
totalWeight=1e3;
%%Begin loop
while abs((unFuelledMass-totalWeight)/totalWeight)>0.001
    unFuelledMass=totalWeight;
    if (verticalTail==1)
        SvTail=0.2*Swing; %Surface of the vertical tail (Sadraey)
    else
        SvTail=0;
    end
    if (horizontalTail==1)
        ShTail=0.2*Swing; %Surface of the vertical tail (Sadraey)
    else
        ShTail=0;
    end

    %%Length
    if nPilots==0
        lCockpit=0;
        lCabin=(floor(nPAX/2)+1)*seatPitch;
    else
        lCockpit=cockpitPitch;
        lCabin=seatPitch*floor((1+nPAX-(2-nPilots))/2);
    end
    if JetEngine==1
        lfueltank=4*fuelWeight/(pi*rhoFuel*dfus^2);
    else
        lfueltank=0;
    end
    Lmeter=rocketEngineLength+lCockpit+lCabin+la+lfueltank+0.5;
    L=Lmeter*3.280840;
    lcyl=(rocketEngineLength+lCabin+lfueltank+0.5)*3.280840;
    Sbody=dfus*lcyl+0.5*la*dfus+0.5*ln*(dfus+db);%Body planform area (ft^2)
    Snose=0.5*dfus*pi*(sqrt((0.5*dfus)^2+la^2));%Nose wetted area (ft^2)
    Scyl=pi*dfus*lcyl;
    Sback=pi*0.5*(dfus+db)*sqrt((dfus-db)^2+ln^2);
    Sfus=Scyl+Snose+Sback; %Fuselage wetted area (ft^2)
    Vcrew=0.8*dfus*pi*(lCabin+lCockpit);

    %%Structural weight
    fuselageWeight=2.167*Sfus^1.075;
    thrustStructureWeight=1.949e-3 * Tr^1.0687;
    noseWeight=Snose*(2.499e-4*qmax +1.7008+ (3.695e-5*qmax-3.252e-3)*dfus);

    %Body weight
    bodyWeight=fuselageWeight+thrustStructureWeight+noseWeight;
    bodyWeight=bodyWeight*(1-TRFbody);

```

```
130     tpsWeight=1.51*Sbody;
131     tpsWeight=tpsWeight*(1-TRFtps);
132
133     if (wing == 1)
134         wingWeight=(Nz*unFuelledMass/(1+eta*Sbody/Swing))^0.386 *...
135             (Swing/troot)^0.572 * (Kwing*b^0.572+Kct*dfus^0.572);
136         wingWeight=wingWeight*(1-TRFwing);
137         Scs=0.05*Swing+0.3*SvTail; %Control surface area (Raymer p.124+125)
138         landingGearWeight=0.010784*unFuelledMass^1.0861+...
139             0.0028*unFuelledMass; %ref 4b+6
140         landingGearWeight=landingGearWeight*(1-TRFlg);
141
142         hTailWeight=Kt*ShTail^1.24; %toBeDone
143         hTailWeight=hTailWeight*(1-TRFtail);
144
145         vTailWeight=Kt*SvTail^1.24;
146         vTailWeight=vTailWeight*(1-TRFtail);
147
148         hydraulicsWeight=Khyd*Scs+1.68e-4*Tr;
149         parachuteWeight=0;
150     else
151         wingWeight=0;
152         Scs=0;
153
154         landingGearWeight=0;
155         vTailWeight=0;
156         hTailWeight=0;
157         hydraulicsWeight=1.68e-4*Tr;
158         parachuteWeight=0.088*unFuelledMass;
159     end
160
161     structuralWeight=wingWeight+bodyWeight+...
162         landingGearWeight+hTailWeight+vTailWeight+tpsWeight+...
163         parachuteWeight;
164
165     %%Equipment weight
166     attitudeControlWeight=1.36e-4*unFuelledMass*L;
167     avionicsWeight=0.055*(unFuelledMass-payloadWeight);
168     avionicsWeight=avionicsWeight*(1-TRFavionics);
169
170     eclssWeight=5.85*Vcrew^0.75+10.9*(nPAX+nPilots)*Ndays+...
171         0.44*avionicsWeight;
172     eclssWeight=eclssWeight*(1-TRFeclss);
173
174     primaryPowerWeight=Kpc*Scs+Kpe*Tr+Kpb*avionicsWeight;
175
176     flightControlWeight=Ksca*Scs+200;
177     electricalWeight=Kecd*unFuelledMass;
178     seatsAndOtherAccessoriesWeight=167*(nPilots+nPAX);
```

```
179
180     equipmentWeight=attitudeControlWeight+avionicsWeight+eclssWeight+...
181         primaryPowerWeight+hydraulicsWeight+flightControlWeight+...
182         electricalWeight+ seatsAndOtherAccessoriesWeight;
183
184     %%Reserve weight
185     reserveWeight=0.05*fuelPropellantWeight;
186
187     %%Total weight
188     totalWeight=structuralWeight+propulsionWeight+payloadWeight+...
189         equipmentWeight+reserveWeight;
190
191 end
192
193 output.EmptyWeight=totalWeight*0.453592; %(kg)
194 output.fuelPropellantWeight=fuelPropellantWeight*0.453592; %(kg)
195 output.length=Lmeter; %(m)
196 output.rocketPerfo=rocketEngineWeightStruct.rocketPerfo; %(SI)
197 output.jetEngineWeight=jetEngineWeight*0.453592; %(kg)
198 output.rocketEngineWeight=rocketEngineWeight*0.453592; %(kg)
199 output.SOtank=rocketEngineWeightStruct.SOtank; %(m^3)
200 output.SPtank=rocketEngineWeightStruct.SPtank; %(m^3)
201 output.jetEngineDiameter=jetEngineDiameter; %(m)
202 output.jetEngineLength=jetEngineLength; %(m)
203 output.jetEngineTSFC=jetEngineTSFC;
204 end
```

### D.2.2 Aerodynamic Module

```
1  function output = aeroLiftModuleSubsonic(M, alpha, Sref, wing, ARwing,...
2      sweepWing, tcWing, TRwing, dfus, db, lcyl, la, ln, CL)
3  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
4  % Aerodynamic module: calculation of the drag coefficient due to lift in
5  % the subsonic regime
6  % Author: Christopher Frank
7  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
8
9  %% Wing
10 if wing==1
11     %Span efficiency factor
12     e = spanEfficiency(sweepWing, ARwing, TRwing, tcWing, M);
13     CdWing=CL^2/(pi*ARwing*e);
14 else
15     CdWing=0;
16 end
17
```

```
%% Fuselage
lfus=la+lcyl+ln;
Sbfus=(pi/4)*db^2; %Fuselage base area (m^2)
Splffus=dfus*0.5*(la+ln+2*lcyl); %Fuselage planform area (m^2)
eta=0.4964*(lfus/dfus)^0.1407;%Surrogate from Matlab
cdc=CdcModel(M,alpha);
CdFuselage=2*alpha^2*Sbfus/Sref + eta*cdc*alpha^3*Splffus/Sref;

%% Total
output=CdWing+CdFuselage;

end
```

```
function output = aeroLiftModuleSupersonic(M, alpha, Sref, wing,...
    ARwing, TRwing, dfus, db, lcyl, la, ln, CL)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Aerodynamic module: calculation of the drag coefficient due to lift in
% the supersonic regime
% Author: Christopher Frank
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%% Wing
if wing==1
    spanWing=sqrt(ARwing*Sref); %Wing span (m)
    rootWing=2*Sref/(spanWing*(1+TRwing)); %Wing root chord (m)
    beta=(M^2-1)^0.5;
    slope=slopeCDLCL2(Sref, spanWing, beta, rootWing, ARwing);
    CdWing=slope*CL^2;

else
    CdWing=0;
end

%% Fuselage
Splffus=dfus*0.5*(la+ln+2*lcyl); %Fuselage planform area (m^2)
Sbfus=(pi/4)*db^2; %Fuselage base area (m^2)
cdc=CdcModel(M, alpha);
CdFuselage=2*alpha^2*Sbfus/Sref+cdc*Splffus*alpha^3/Sref;

%Output
output=CdWing+CdFuselage;
end
```

```
function output = aeroModule(Sref, wing, ARwing, sweepWing, tcWing,...
    TRwing, dfus, db, lcyl, la, ln, sweepHTail, ARHTail, dnac, lnac,...
    JetEngine, sweepVTail, ARVTail, horizontalTail, verticalTail)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Aerodynamic module: main file for the calculation of the aerodynamic
% coefficients
% Author: Christopher Frank
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%% Calculation of the coefficents for future regressions
Cd06h0=aeroModuleSubsonic(0, 0.6, Sref, wing, ARwing, sweepWing,...
    tcWing, TRwing, dfus, db, lcyl, la, ln, sweepHTail, ARHTail,...
    dnac, lnac, JetEngine, sweepVTail, ARVTail,...
    horizontalTail, verticalTail);
Cd06h1=aeroModuleSubsonic(20000, 0.6, Sref, wing, ARwing, sweepWing,...
    tcWing, TRwing, dfus, db, lcyl, la, ln, sweepHTail, ARHTail,...
    dnac, lnac, JetEngine, sweepVTail, ARVTail,...
    horizontalTail, verticalTail);
Cd12h0=aeroModuleSupersonic(0, 1.2, Sref, wing, ARwing, sweepWing,...
    tcWing, TRwing, dfus, db, lcyl, la, ln, sweepHTail, ARHTail,...
    dnac, lnac, JetEngine, sweepVTail, ARVTail,...
    horizontalTail, verticalTail);
Cd12h1=aeroModuleSupersonic(20000, 1.2, Sref, wing, ARwing,...
    sweepWing, tcWing, TRwing, dfus, db, lcyl, la, ln, sweepHTail,...
    ARHTail, dnac, lnac, JetEngine, sweepVTail, ARVTail,...
    horizontalTail, verticalTail);
Cd3h0=aeroModuleSupersonic(0, 3, Sref, wing, ARwing, sweepWing,...
    tcWing, TRwing, dfus, db, lcyl, la, ln, sweepHTail, ARHTail,...
    dnac, lnac, JetEngine, sweepVTail, ARVTail,...
    horizontalTail, verticalTail);
Cd3h1=aeroModuleSupersonic(20000, 3, Sref, wing, ARwing, sweepWing,...
    tcWing, TRwing, dfus, db, lcyl, la, ln, sweepHTail, ARHTail,...
    dnac, lnac, JetEngine, sweepVTail, ARVTail,...
    horizontalTail, verticalTail);
Cd5h0=aeroModuleSupersonic(0, 5, Sref, wing, ARwing, sweepWing,...
    tcWing, TRwing, dfus, db, lcyl, la, ln, sweepHTail, ARHTail,...
    dnac, lnac, JetEngine, sweepVTail, ARVTail,...
    horizontalTail, verticalTail);
Cd5h1=aeroModuleSupersonic(20000, 5, Sref, wing, ARwing, sweepWing,...
    tcWing, TRwing, dfus, db, lcyl, la, ln, sweepHTail, ARHTail,...
    dnac, lnac, JetEngine, sweepVTail, ARVTail,...
    horizontalTail, verticalTail);
Cd7h0=aeroModuleSupersonic(0, 7, Sref, wing, ARwing, sweepWing,...
    tcWing, TRwing, dfus, db, lcyl, la, ln, sweepHTail, ARHTail,...
    dnac, lnac, JetEngine, sweepVTail, ARVTail,...
    horizontalTail, verticalTail);
Cd7h1=aeroModuleSupersonic(20000, 7, Sref, wing, ARwing, sweepWing,...
    tcWing, TRwing, dfus, db, lcyl, la, ln, sweepHTail, ARHTail,...
    dnac, lnac, JetEngine, sweepVTail, ARVTail,...
```

```
    horizontalTail, verticalTail);

output.Cd06=[Cd06h0 Cd06h1];
output.Cd12=[Cd12h0 Cd12h1];

Cd1h0=Cd06h0+0.5*(Cd12h0-Cd06h0);
Cd1h1=Cd06h1+0.5*(Cd12h1-Cd06h1);
output.Cd1=[Cd1h0 Cd1h1];

a0 = 6.25*Cd1h0-6.25*Cd06h0;
b0 = -7.5*Cd1h0 +7.5*Cd06h0;
c0 = 2.25*Cd1h0 -1.25*Cd06h0;

a1 = 6.25*Cd1h1-6.25*Cd06h1;
b1 = -7.5*Cd1h1 +7.5*Cd06h1;
c1 = 2.25*Cd1h1 -1.25*Cd06h1;

output.coef=[a0 b0 c0 ; a1 b1 c1];

Cd1h0=Cd06h0+0.5*(Cd12h0-Cd06h0);
Cd1h1=Cd06h1+0.5*(Cd12h1-Cd06h1);

output.Cd1=[Cd1h0 Cd1h1];

aa0 = -17.77777778*Cd12h0;
bb0 = 40.*Cd12h0;
cc0 =-21.40000000*Cd12h0;

aa1 = -17.77777778*Cd12h1;
bb1 = 40.*Cd12h1;
cc1 =-21.40000000*Cd12h1;

output.coefcoef=[aa0 bb0 cc0 ; aa1 bb1 cc1];

A0=-(.2000000000*(-522.*Cd3h0*Cd12h0*Cd7h0+171.*Cd3h0*Cd12h0*Cd5h0-...
    200.*Cd3h0*Cd7h0*Cd5h0+551.*Cd7h0*Cd12h0*Cd5h0))/(-38.*Cd3h0*Cd7h0...
    +29.*Cd12h0*Cd7h0+9.*Cd7h0*Cd5h0+9.*Cd12h0*Cd3h0+29.*Cd3h0*Cd5h0-...
    38.*Cd12h0*Cd5h0);
B0=(.6000000000*(-870.*Cd3h0*Cd12h0*Cd7h0+399.*Cd3h0*Cd12h0*Cd5h0-...
    80.*Cd3h0*Cd7h0*Cd5h0+551.*Cd7h0*Cd12h0*Cd5h0))/(-38.*Cd3h0*Cd7h0+...
    29.*Cd12h0*Cd7h0+9.*Cd7h0*Cd5h0+9.*Cd12h0*Cd3h0+29.*Cd3h0*Cd5h0-...
    38.*Cd12h0*Cd5h0);
C0=-(.2000000000*(1189.*Cd12h0*Cd7h0+1160.*Cd3h0*Cd5h0-1900.*Cd3h0*...
    Cd7h0+189.*Cd12h0*Cd3h0-1178.*Cd12h0*Cd5h0+540.*Cd7h0*Cd5h0))/...
    (-38.*Cd3h0*Cd7h0+29.*Cd12h0*Cd7h0+9.*Cd7h0*Cd5h0+9.*Cd12h0*Cd3h0+...
    29.*Cd3h0*Cd5h0-38.*Cd12h0*Cd5h0);
D0=(.6000000000*(-1330.*Cd3h0*Cd7h0+54.*Cd12h0*Cd3h0+725.*Cd3h0*Cd5h0+...
    406.*Cd12h0*Cd7h0-380.*Cd12h0*Cd5h0+525.*Cd7h0*Cd5h0))/(-38.*Cd3h0*...
    Cd7h0+29.*Cd12h0*Cd7h0+9.*Cd7h0*Cd5h0+9.*Cd12h0*Cd3h0+29.*Cd3h0*...
```

```
    Cd5h0-38.*Cd12h0*Cd5h0);

A1=-(.2000000000*(-522.*Cd3h1*Cd12h1*Cd7h1+171.*Cd3h1*Cd12h1*Cd5h1-...
    200.*Cd3h1*Cd7h1*Cd5h1+551.*Cd7h1*Cd12h1*Cd5h1))/(-38.*Cd3h1*Cd7h1+...
    29.*Cd12h1*Cd7h1+9.*Cd7h1*Cd5h1+9.*Cd12h1*Cd3h1+29.*Cd3h1*Cd5h1-...
    38.*Cd12h1*Cd5h1);
B1=(.6000000000*(-870.*Cd3h1*Cd12h1*Cd7h1+399.*Cd3h1*Cd12h1*Cd5h1-...
    80.*Cd3h1*Cd7h1*Cd5h1+551.*Cd7h1*Cd12h1*Cd5h1))/(-38.*Cd3h1*Cd7h1+...
    29.*Cd12h1*Cd7h1+9.*Cd7h1*Cd5h1+9.*Cd12h1*Cd3h1+29.*Cd3h1*Cd5h1-...
    38.*Cd12h1*Cd5h1);
C1=-(.2000000000*(1189.*Cd12h1*Cd7h1+1160.*Cd3h1*Cd5h1-1900.*Cd3h1*Cd7h1...
    +189.*Cd12h1*Cd3h1-1178.*Cd12h1*Cd5h1+540.*Cd7h1*Cd5h1))/...
    (-38.*Cd3h1*Cd7h1+29.*Cd12h1*Cd7h1+9.*Cd7h1*Cd5h1+9.*Cd12h1*Cd3h1+...
    29.*Cd3h1*Cd5h1-38.*Cd12h1*Cd5h1);
D1=(.6000000000*(-1330.*Cd3h1*Cd7h1+54.*Cd12h1*Cd3h1+725.*Cd3h1*Cd5h1+...
    406.*Cd12h1*Cd7h1-380.*Cd12h1*Cd5h1+525.*Cd7h1*Cd5h1))/(-38.*Cd3h1*...
    Cd7h1+29.*Cd12h1*Cd7h1+9.*Cd7h1*Cd5h1+9.*Cd12h1*Cd3h1+29.*Cd3h1*...
    Cd5h1-38.*Cd12h1*Cd5h1);

output.COEF= [A0 B0 C0 D0 ; A1 B1 C1 D1];

end
```

```
function output = aeroModuleSubsonic(h, M, Sref, wing, ARwing,...
    sweepWing, tcWing, TRwing, dfus, db, lcyl, la, ln, sweepHTail,...
    ARHTail, dnac, lnac, jetEngine, sweepVTail, ARVTail, horizontalTail,...
    verticalTail)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Aerodynamic module: calculation of the drag coefficient in the subsonic
% regime
% Author: Christopher Frank
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%% Flight conditions
atm=funcAtm(h);
Tinf=atm.T; %Function of h
V=MtoV(M,h);

%% Wing
if wing==1
    %Parameters
    Lprime=1.2; %Constant defined by Roskam
    SwetWing=2*Sref; %Wing swept area (m2)
    spanWing=sqrt(ARwing*Sref); %Wing span (m)
    rootWing=2*Sref/(spanWing*(1+TRwing)); %Wing root chord (m)
    %Wing mean chord (m)
```

```
    meanChordWing=rootWing*2*(1+TRwing+TRwing^2)/(3*(1+TRwing));
    ReW=Reynolds (MtoV(M,h), meanChordWing, h); %Wing Reynolds number
    Rwffus= Rwf(M, ReW);
    Rlsfus=Rls( M, sweepWing );
    Cfw= Cf( M, Tinf, ReW );
    CdWing=Rwffus*Rlsfus*Cfw*(1+Lprime*tcWing+100*tcWing^4)*SwetWing/Sref;

else
    CdWing=0;
end

%% Fuselage
lfus=la+lcyl+ln;
apothemFront=sqrt(0.25*dfus^2+la^2); %Length of the front apothem (m)
apothemBack=sqrt(0.25*dfus^2+ln^2); %Length of the back apothem (m)
%Fuselafe wetted area (m^2)
Swetfus=pi*dfus*0.5*(apothemFront+apothemBack+2*lcyl);
Sbfus=(pi/4)*db^2; %Fuselage base area (m^2)
Sfus=Sbfus; %Fuselage maximum front area (m^2)
ReFus=Reynolds (V, lfus, h); %Fuselage Reynolds number

%Zero-lift drag coefficient
if wing == 1
    Rwffus= Rwf(M, ReFus);
elseif wing == 0
    Rwffus=1;
end

Cffus= Cf(M, Tinf, ReFus);
Cdb0fus=Rwffus*Cffus*(1+60*(lfus/dfus)^(-3)+0.0025*(lfus/dfus))*...
    Swetfus/Sref;
Cdbfus=0.029*(db/dfus)^3 * (Cdb0fus*Sref/Sfus)^(-0.5)  * Sfus/Sref;
CdFuselage=Rwffus*Cffus*(1+60*(lfus/dfus)^(-3)+0.0025*(lfus/dfus))*...
    Swetfus/Sref+Cdbfus;

if verticalTail==1
    %Vertical tail
    cdvtail = empennageDragSubsonic(M, h, Sref, ARVTail, Tinf, sweepVTail);
    Cd0VerticalTail = cdvtail;
else
    Cd0VerticalTail=0;
end

if horizontalTail==1
    %Horizontal tail
    cdhtail = empennageDragSubsonic(M, h, Sref, ARHTail, Tinf, sweepHTail);
    Cd0HorizontalTail = cdhtail;
else
    Cd0HorizontalTail=0;
```

```
end

%% Nacelle
if jetEngine==1
    ReNac=Reynolds (MtoV(0.6,h), lnac, h); %Nacelle Reynolds number
    fnac=lnac/dnac;
    FFnac=1+0.35/fnac;
    Cfnac= Cf( M,Tinf,ReNac );
    Qnacelle=1.5; %Raymer
    SwetNacelle=pi*dnac*lnac;
    CdNacelle=Cfnac*FFnac*Qnacelle*SwetNacelle/Sref;
else
    CdNacelle=0;
end

%% Total
Cd0Vehicle=CdWing+CdFuselage+Cd0VerticalTail+...
    Cd0HorizontalTail+CdNacelle;
CdVehicle=Cd0Vehicle;

output=CdVehicle;

end
```

```
function output = aeroModuleSupersonic(h, M, Sref, wing, ARwing,...
    sweepWing, tcWing, TRwing, dfus, db, lcyl, la, ln, sweepHTail,...
    ARHTail, dnac, lnac, jetEngine, sweepVTail, ARVTail, horizontalTail,...
    verticalTail)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Aerodynamic module: calculation of the drag coefficient in the supersonic
% regime
% Author: Christopher Frank
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%% Flight conditions
atm=funcAtm(h);
Tinf=atm.T; %Function of h
V=MtoV(M,h);

%% Wing
if wing==1
    SwetWing=2*Sref; %Wing swept area (m2)
    spanWing=sqrt(ARwing*Sref); %Wing span (m)
    rootWing=2*Sref/(spanWing*(1+TRwing)); %Wing root chord (m)
    %Wing mean chord (m)
    meanChordWing=rootWing*2*(1+TRwing+TRwing^2)/(3*(1+TRwing));
```

```
    tipWing=TRwing*rootWing; %Wing tip chord (m)
    %https://www.physicsforums.com/threads/airfoil-leading-edge-radius
    rBE=1.1019*tcWing^2;
    ReW=Reynolds (V, meanChordWing, h); %Wing Reynolds number

    Cfw=Cf( M, Tinf, ReW );
    Cdwf=Cfw*SwetWing/Sref;
    CdLEw= CdLE( M,sweepWing, rBE, Sref, spanWing );
    tceff=tceffCalc( Sref, spanWing, tcWing, tipWing, rootWing);
    Sbw=Sref;
    beta=(M^2-1)^0.5;
    Cdwave=CdLEw+16*tceff^2/(3*beta)*Sbw/Sref;
    CdWing=Cdwf+Cdwave;

else
    CdWing=0;
end

%% Fuselage
lfus=la+lcyl+ln;
apothemFront=sqrt(0.25*dfus^2+la^2); %Length of the front apothem (m)
apothemBack=sqrt(0.25*dfus^2+ln^2); %Length of the back apothem (m)
%Fuselage wetted area (m^2)
Swetfus=pi*dfus*0.5*(apothemFront+apothemBack+2*lcyl);
Sbfus=(pi/4)*db^2; %Fuselage base area (m^2)
Sfus=Sbfus; %Fuselage maximum front area (m^2)
ReFus=Reynolds (V, lfus, h); %Fuselage Reynolds number
dn=0; %Nose diameter = 0

CdN2 = CdNModel(M, la, dfus, dn); %Front part
CdA = CdNModel(M, ln, dfus, dn); %Rear part
Cffus= Cf(M, Tinf, ReFus);
Cdbfus= CdbfusModel(M);
CdNANC= CdNANCModel(la, ln, lcyl, dfus);
CdFuselage=Cffus*Swetfus/Sref + (CdN2+CdA+Cdbfus+CdNANC)*Sfus/Sref;

%% Vertical tail
if verticalTail==1

    cdvtail = empennageDragSupersonic(M, V, h, Sref, ARVTail, Tinf,...
        sweepVTail);
    Cd0VerticalTail = cdvtail;
else
    Cd0VerticalTail=0;
end

%% Horizontal tail
if horizontalTail==1

```

```
    cdhtail = empennageDragSupersonic(M, V, h, Sref, ARHTail, Tinf,...
        sweepHTail);
    Cd0HorizontalTail = cdhtail;
else
    Cd0HorizontalTail=0;
end

%% Nacelle
if jetEngine==1
    ReNac=Reynolds (MtoV(0.6,h), lnac, h); %Nacelle Reynolds number
    fnac=lnac/dnac;
    FFnac=1+0.35/fnac;
    Cfnac= Cf( M,Tinf,ReNac );
    Qnacelle=1.5; %Raymer
    SwetNacelle=pi*dnac*lnac;

    CdNacelle=Cfnac*FFnac*Qnacelle*SwetNacelle/Sref;

else
    CdNacelle=0;
end

Cd0Vehicle=CdWing+CdFuselage+Cd0VerticalTail+...
    Cd0HorizontalTail+CdNacelle;
CdVehicle=Cd0Vehicle;

output=CdVehicle;

end
```

```
function output = aeroTOLAModule(Wto, Wlanding, CL, Sref, TOmode, LAmode)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Aerodynamic module: calculation of the drag related to take-off and
% landing
% Author: Christopher Frank
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

deflectionFlaps=0.52; %Deflection = 30  as provided by Sadraey

%% Landing gear
if TOmode == 1 %Horizontal TO
    WlgCalc=Wto;
    cdgear = landingGearDrag( CL, Sref, WlgCalc );
    CdLandingGear = cdgear;
elseif ((LAmode == 0) || (LAmode == 1)) %Horizontal landing
    WlgCalc=Wlanding;
```

```
    cdgear = landingGearDrag( CL, Sref, WlgCalc );
    CdLandingGear = cdgear;
else %Vertical TO and landing
    CdLandingGear = 0;
end

%% Flaps
CdFlaps = flapsDrag(deflectionFlaps);
CdTOandLA=CdLandingGear+CdFlaps;

output=CdTOandLA;

end
```

```
function output = aeroTrajectory(M, h, Cd06, Cd12,...
    coef, Cd1, coefcoef, COEF)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Aerodynamic module: calculation of the vehicle drag coefficient for the
% trajectory module
% Author: Christopher Frank
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

Cd1h0=Cd1(1);
Cd1h1=Cd1(2);

Cd12h0=Cd12(1);
Cd12h1=Cd12(2);

Cd06h0=Cd06(1);
Cd06h1=Cd06(2);

a0=coef(1,1);
b0=coef(1,2);
c0=coef(1,3);
a1=coef(2,1);
b1=coef(2,2);
c1=coef(2,3);

aa0=coefcoef(1,1);
bb0=coefcoef(1,2);
cc0=coefcoef(1,3);
aa1=coefcoef(2,1);
bb1=coefcoef(2,2);
cc1=coefcoef(2,3);

A0=COEF(1,1);
```

```
B0=COEF(1,2);
C0=COEF(1,3);
D0=COEF(1,4);
A1=COEF(2,1);
B1=COEF(2,2);
C1=COEF(2,3);
D1=COEF(2,4);

%% Mach number dependence
if M<=0.6
    Cdh0=Cd06h0;
    Cdh1=Cd06h1;

elseif M<=1 %Parabolic interpolation between M=0.6 and M=1
    Cdh0=a0*M^2+b0*M+c0;
    Cdh1=a1*M^2+b1*M+c1;

elseif M<=1.05 %Linear interpolation between M=1 and M=1.05
    Cdh0=(-20*Cd1h0+20*Cd12h0)*M+21*Cd1h0-20*Cd12h0;
    Cdh1=(-20*Cd1h1+20*Cd12h1)*M+21*Cd1h1-20*Cd12h1;

elseif M<=1.2 %Parabolic interpolation between M=1.05 and M=1.2
    Cdh0=aa0*M^2+bb0*M+cc0;
    Cdh1=aa1*M^2+bb1*M+cc1;

else
    Cdh0=(A0*M+B0)/(M^2+C0*M+D0);
    Cdh1=(A1*M+B1)/(M^2+C1*M+D1);
end

%% Altitude dependence
aAlt=(Cdh0-Cdh1)/(-20000);
bAlt=Cdh0;
CdVehicle=aAlt*h+bAlt;

output=CdVehicle;
end
```

```
function output = CdbfusModel(M)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Aerodynamic module: calculation of CDbfus
% Author: Christopher Frank
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

a =      0.2508;
b =     -0.5768;
```

```
c =       0.104;
d =     -0.2178;

output = a*exp(b*M) + c*exp(d*M);

end
```

```
function output = CdcModel(M,alpha)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Aerodynamic module: calculation of Cdc
% Author: Christopher Frank
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

Mc=M*sin(alpha);

if Mc <1 %Subsonic
    %Coefficients for subsonic speed
    a1=0.1986; b1=1.05; c1=0.121; %1st term coefs
    a2=0.567; b2=0.8343; c2=0.2867; %2nd term coefs
    a3=7.182; b3=218.3; c3=163.2; %3rd term coefs

    output=a1*exp(-((Mc-b1)/c1)^2)  +a2*exp(-((Mc-b2)/c2)^2)+ ...
        a3*exp(-((Mc-b3)/c3)^2);

else %Supersonic
    a1 = 0.2642 ; b1 =1.024;  c1 = 0.3659; %1st term coefs
    a2 =3.87;  b2 =9.531 ;  c2 = 8.874 ; %2nd term coefs

    output=a1*exp(-((Mc^(-1)-b1)/c1)^2)  +a2*exp(-((Mc^(-1)-b2)/c2)^2);

end

end
```

```
function output = CdLE( M,lamndatcMax,rBE,Sref,span )
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Aerodynamic module: calculation of CdLE
% Author: Christopher Frank
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

temp=1.28*(M^3*cos(lamndatcMax)^6)/(1+(M*cos(lamndatcMax))^3);
frac=Sref/(2*rBE*span/cos(lamndatcMax));
```

```

output=temp/frac;

end
```

```
function output = CdNANCModel(la, ln, lcyl,df)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Aerodynamic module: calculation of Cda(NC)
% Author: Christopher Frank
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

X=min(lcyl/la,1.8);
Z=ln/la;

temp=0.690420248702588 + -0.689907629817987 * Z + -1.48590778036454 * X...
    + (Z - 1.15384615384615) * (Z - 1.15384615384615) * 0.281155693559382...
    +  (Z - 1.15384615384615) * (X - 0.600903625630933) * ...
    0.334536224153579 + (X -    0.600903625630933) *...
    (X - 0.600903625630933) * 0.613753529697789;

output=temp/((2*la/df)^2);

end
```

```
function output = CdNModel(M, lforaft, df, dn)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Aerodynamic module: calculation of CdN
% Author: Christopher Frank
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

X=(dn/df)^2;
beta=sqrt(M^2-1);
Z=2*lforaft/(df*beta);

temp=2.27914395837144 + 0.0621279592946248 * Z + -4.29374905998544 *...
    X + (Z - 9.22702702702703) * ((Z - 9.22702702702703) *...
    -0.00433046083467327) + (Z -9.22702702702703) *...
    ((X - 0.386891710261467) * -0.27048494542039) + (X -...
    0.386891710261467) * ((X - 0.386891710261467) * 6.81231966453556)...
    + (Z -9.22702702702703) * ((Z - 9.22702702702703) *...
    ((X - 0.386891710261467) *0.0119444696425639)) + ...
    (Z - 9.22702702702703) * ((X - 0.386891710261467) * ((X -...
```

```
    0.386891710261467) * 0.286581345990331)) + (X - 0.386891710261467)...
    * ((X -0.386891710261467) * ((X - 0.386891710261467)...
    * -5.77828976074559));

output=max(temp,0)/((2*lforaft/df)^2);
end
```

```
function output = Cf( M,Tinf,Re )
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Aerodynamic module: calculation of Cf
% Author: Christopher Frank
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%Cf in incompressible
cfinc=0.074*Re.^(-0.2);

%Definition of intermediate variables
T_Tinf=1+0.1151.*M.^2;
R_Rinf=T_Tinf.^(-2.5).*(T_Tinf+216./Tinf)./(1+216./Tinf);

%Ratio calculation
cf_cfinc=T_Tinf.^(-1).*R_Rinf.^(-0.2);

%Output assignment
output=cf_cfinc*cfinc;

end
```

```
function output = empennageDragSubsonic(M, h, Sref, AREmp, Tinf, sweepEmp)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Aerodynamic module: calculation of the drag coefficient due to lift in
% the subsonic regime
% Author: Christopher Frank
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%Calculate the drag coefficient of the empennage

%Design variables
SEmp=0.2*Sref;
tcEmp=0.08;
TREmp=0.35;
SwetEmp=2*SEmp; %Wing swept area (m2)
spanEmp=sqrt(AREmp*SEmp); %Wing span (m)
```

```
rootEmp=2*SEmp/(spanEmp*(1+TREmp)); %Wing root chord (m)
meanChordEmp=rootEmp*2*(1+TREmp+TREmp^2)/(3*(1+TREmp));%Wing mean chord (m)
Lprime=1.2; %Constant defined by Roskam

ReEmp=Reynolds (MtoV(0.6,h), meanChordEmp, h); %Fuselage Reynolds number
Rwfemp= Rwf(M, ReEmp);
Rlsemp=Rls(M, sweepEmp);
Cfe= Cf(M, Tinf, ReEmp);
output=Rwfemp*Rlsemp*Cfe*(1+Lprime*tcEmp+100*tcEmp^4)*SwetEmp/Sref;


end
```

```
function output = empennageDragSupersonic(M, V, h, Sref, AREmp, Tinf,...
    sweepEmp)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Aerodynamic module: calculation of the drag coefficient of the empennage
% in the supersonic regime
% Author: Christopher Frank
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%Design variables
SEmp=0.2*Sref;
tcEmp=0.08;
TREmp=0.35;
SwetEmp=2*SEmp; %Wing swept area (m2)
spanEmp=sqrt(AREmp*SEmp); %Wing span (m)
rootEmp=2*SEmp/(spanEmp*(1+TREmp)); %Wing root chord (m)
meanChordEmp=rootEmp*2*(1+TREmp+TREmp^2)/(3*(1+TREmp));%Wing mean chord (m)
tipEmp=TREmp*rootEmp; %Wing tip chord (m)
%https://www.physicsforums.com/threads/airfoil-leading-edge-radius.507700/
rBE=1.1019*tcEmp^2;
ReEmp=Reynolds (V, meanChordEmp, h); %Fuselage Reynolds number

Cfemp=Cf(M, Tinf, ReEmp);
Cdempf=Cfemp*SwetEmp/Sref;

CdLEemp= CdLE(M, sweepEmp, rBE, Sref, spanEmp);
tceff=tceffCalc(SEmp, spanEmp, tcEmp, tipEmp, rootEmp);
beta=(M^2-1)^0.5;
Cdwave=CdLEemp+16*tceff^2/(3*beta)*SEmp/Sref;
output=Cdempf+Cdwave;

end
```

```
function output = flapsDrag(deflectionFlaps)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Aerodynamic module: calculation of the drag coefficient due to flaps
% Author: Christopher Frank
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

cf_c=0.3; %flap chord over wing chord as provided by Sadraey
bf_b=0.6; %flap span over wing span as provided by Sadraey
Sflap_Sref=bf_b*cf_c;
output=0.9*(cf_c)^1.38*(Sflap_Sref)*(sin(deflectionFlaps)^2);

end
```

```
function output = landingGearDrag( CL, Sref, W )
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Aerodynamic module: calculation of the drag coefficient of the landing
% gear
% Author: Christopher Frank
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%Main landing gear
Cd_gearm_cl0=1.3;
pm=-0.4*Cd_gearm_cl0;
Dm=0.01*5.1*(0.9*W/4)^0.302;
bm=0.01*0.36*(0.45*W)^0.467;
Sgearm=Dm*bm;
Cd_maingear=(Cd_gearm_cl0+pm*CL)*Sgearm/Sref;

%Nose landing gear
Cd_gearn_cl0=0.5;
pn=-0.25*Cd_gearn_cl0;
Dn=0.6*Dm;
bn=0.6*bm;
Sgearn=Dn*bn;
Cd_nosegear=(Cd_gearn_cl0+pn*CL)*Sgearn/Sref;

output=Cd_maingear+Cd_nosegear;

end
```

```
function output = liftSlope(AR, sweepWing, Sref, M, tcWing, dfus, b, rootChord)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
% Aerodynamic module: calculation of the lift slope
% Author: Christopher Frank
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

etaLift=0.95; %Raymer p.312
F=1.07*((1+dfus/b)^2);
Mdd=0.786243011114196 + 0.00361901304093074 * sweepWing ...
    -0.643773595710193 * tcWing + (sweepWing - 40.9964954704591) * ...
    ((sweepWing - 40.9964954704591) * 0.0000257434767091937)...
    + (sweepWing - 40.9964954704591) * ((tcWing - 0.0801980198019802) *...
    0.0102064727164255);

Sexposed=Sref-dfus*rootChord;
if M<Mdd
    beta2=sqrt(1-M^2);
    endEquation= min(0.98,Sexposed*F/Sref);
    denom=2+sqrt(4+(AR*beta2/etaLift)^2*(1+((tan(sweepWing))^2)/beta2^2));
    slope=2*pi*AR*endEquation/denom;
elseif M>1.2
    slopeMdd=liftSlope(AR, sweepWing, Sref, Mdd*0.9999, Mdd, dfus, b,...
        rootChord);
    correctionFactor=0.85*slopeMdd/6.03;
    betaFac=sqrt(M^2-1);
    slope=4*correctionFactor/betaFac;
else
    %Fitting with Matlab: curve goes through Mdd, 1.2 and max at 1.1 of
    %yMdd to match curves from Raymer
    yMdd=liftSlope(AR, sweepWing, Sref, Mdd*0.9999, Mdd, dfus, b,...
        rootChord);
    yM12=0.85*liftSlope(AR, sweepWing, Sref, Mdd*0.9999, Mdd, dfus, b,...
        rootChord);
    yMax=1.1*yMdd;
    a=-(22.72727273*(60.*yMax-11.*yMdd-50.*Mdd*yMax-49.*yM12+50.*...
        Mdd*yM12))/(250.*Mdd^2+294.-545.*Mdd);
    b=(.4545454545*(3600.*yMax-1199.*yMdd-2401.*yM12-2500.*yMax*Mdd^2+...
        2500.*Mdd^2*yM12))/(250.*Mdd^2+294.-545.*Mdd);
    c=-(0.9090909091e-1*(-3234.*yMdd+18000.*Mdd*yMax-12005.*Mdd*yM12-...
        15000.*yMax*Mdd^2+12250.*Mdd^2*yM12))/(250.*Mdd^2+294.-545.*Mdd);
    slope=a*M^2+b*M+c;
end
output=slope;
end
```

```
function output = Reynolds (V, l, h)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Aerodynamic module: calculation of the Reynolds number
```

```
% Author: Christopher Frank
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

atm=funcAtm(h);%Calculate atmospheric parameter as a function of altitude

mu=atm.mu; %Dynamic viscosity
rho=atm.rho; %Density

output=rho*V*l/mu;%Reynolds number

end
```

```
function output = Rls( M, lamndatcMax )
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Aerodynamic module: calculation of Rls
% Author: Christopher Frank
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%Intermediate variables
cosL=cos(lamndatcMax);
Mmax=max(0.25,M);

%Variable assignment
output=0.322709056484223 + 0.572331377882922 * cosL + 0.521452969288079...
    * Mmax + (cosL - 0.74241582759369) * (cosL - 0.74241582759369) *...
    -0.892082739967756 + (Mmax - 0.641406249999999) *...
    (Mmax - 0.641406249999999) * 0.720296196196063;

end
```

```
function output = Rwf( M, Re )
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Aerodynamic module: calculation of Rwf
% Author: Christopher Frank
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%Intermediate variables
Mmax=max(0.25,M);
logRe=log(Re);

%Variable assignment
output=1.06499242343811 + -0.0791799780084577 * Mmax + (logRe -...
```

```
    17.4180492814283 ) * (logRe - 17.4180492814283) * ...
    -0.00368622098768004 + ( logRe - 17.4180492814283) * (Mmax -...
    0.594927536231885) * 0.15436730244968 + (Mmax - 0.594927536231885)...
    * (Mmax - 0.594927536231885) * -0.346130269968011 + ( logRe -...
    17.4180492814283) * (logRe - 17.4180492814283) * (...
    logRe - 17.4180492814283) * 0.00102734087400994 + (logRe...
    - 17.4180492814283) * (logRe - 17.4180492814283) * (logRe...
    - 17.4180492814283) * (Mmax - 0.594927536231885) *...
    -0.0171779814633487 + (Mmax -  0.594927536231885) * (Mmax -...
    0.594927536231885) * (Mmax - 0.594927536231885) * (...
    logRe - 17.4180492814283) * 0.417680110192136;

end
```

```
function output = slopeCDLCL2(Sref, span, beta, croot, AR)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Aerodynamic module: calculation of the slope CDL over CL2
% Author: Christopher Frank
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

p=Sref/(span*croot);
ps=beta*span/(2*croot);

temp=0.9333*ps+0.23;

if temp < 0.55
    temp=0.55;
end

output=temp/(pi*AR*p/(1+p));

end
```

```
function output = spanEfficiency(sweep, AR, TR, tc, M)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Aerodynamic module: calculation of the span efficiency
% Author: Christopher Frank
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%flambda
flambda=0.005*(1+1.5*(TR-0.6)^2);

```

```
temp=1+(0.142+flambda*AR*(10*tc)^0.33)/((cos(sweep))^2) +...
    0.1/((4+AR)^(0.8));

%Output assignment
output=((1+0.12*M^6)*temp)^(-1);

end
```

```
function output = tceffCalc(Sref, span, tc, ctip, croot)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Aerodynamic module: calculation of t_ceff based on a linear evolution
% of the chord from croot (y=0) to ctip (y=b/2): a*y+b
% Author: Christopher Frank
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

bb=croot;
aa=2*(ctip-croot)/span;

x0=0;
xEnd=span/2;

int=(2/(3*aa))* (  (aa*xEnd+bb)^(3/2) - (aa*x0+bb)^(3/2)   );

output=int*tc*2/Sref;

end
```

### D.2.3 Economic Module

```
function output=costModule(wing, nLaunch, programLength, numbUnits,...
    Tmiss, nPilots, nPAX, TOmode, year, LC, nJet, JetEngine, Isp,...
    propellantWeight, Propellant, O_F, Tr, SOtank, SPtank,...
    TotalLaunchMass, VehicleDryMass, rocketEngineEmptyWeight,...
    afterburner, WdryjetEng, TIT, Tj, fuelWeight)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Cost module: function that drives the cost calculation
% Author: Christopher Frank
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%% Jet engine cost
if JetEngine==1
    jetEngineCost=JetEngineCost(TIT, WdryjetEng, LC, afterburner,...
        year, Tmiss, nJet, Tj, fuelWeight)
```

```
else
    jetEngineCost.dev=0; jetEngineCost.prod=0; jetEngineCost.maint=0;...
        jetEngineCost.propellant=0;
end

%% Rocket engine cost
%Constants
Q=1631;%Total number of engines produced (middle of the range) //Solid
Nn=1; % Number of nozzles //Solid
Proto=56; %Number of prototypes //Both
rocketEngineCost=rocketCostModule(Q, Nn, LC, Proto,...
    rocketEngineEmptyWeight, Isp, propellantWeight, year, Propellant,...
    O_F, Tr, SOtank, SPtank);

%% Total cost
output=costModuleAirframe(LC, wing, nLaunch, programLength, numbUnits,...
    Tmiss, nPilots, nPAX, TOmode, year, nJet,...
    WdryjetEng, propellantWeight, rocketEngineEmptyWeight,...
    VehicleDryMass, TotalLaunchMass, jetEngineCost, rocketEngineCost, ...
    O_F, Propellant);

end
```

```
function output=costModuleAirframe(LC, wing, nLaunch, programLength,...
    numbUnits, Tmiss, nPilots, nPAX, TOmode, year, nJet,...
    jetEngineEmptyWeight, mProp, engineEmptyWeight, VehicleDryMass,...
    TotalLaunchMass, jetEngineCost, rocketEngineCost, O_F, Propellant)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Cost module: function that calculates the total cost
% Author: Christopher Frank
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%% Constants
Vw=5; %Maximum Mach Number
nRocket=1; %Number of rocket engines
stageNumber=1; %Number of vehicle stage

%% Constants for cost calculation
%Number of reuses for each component
reuse.Jet=200;
reuse.Body=200;
reuse.Rocket=200;

%% Rates definition for cost calculation
rates.IR=0.02; %Interest rate
rates.tIR=10; %Years of interest rate
```

```
rates.RR=0.05; %Repayment rate
rates.tRR=10; %Years of repayment rate
rates.Rabort=1/40; %Insurance abort rate
rates.Rloss=1/2000; %Insurance loss rate

%% General factors description
generalFactors.fPS=1.075;%Project System Engineering Factor
generalFactors.fTD=1;%Technical development factor

%Technical quality factor
generalFactors.fTQ.rocket=1;
generalFactors.fTQ.winged=Vw^0.16;
generalFactors.fTE=1;%Team Experience Factor
generalFactors.fI=1.025;%Integration Factors

%Cost Reduction Factor
LCexp=log(0.01*LC)/log(2);
generalFactors.fCR.rocket=1.004*((numbUnits*nRocket)^(LCexp));
generalFactors.fCR.jet=1.004*((numbUnits*nJet)^(LCexp));
generalFactors.fCR.ballistic=1.004*((numbUnits)^(LCexp));
generalFactors.fCR.winged=1.004*((numbUnits)^(LCexp));

%Commercial factor
generalFactors.fC.rocket=0.2;
generalFactors.fC.jet=1;
generalFactors.fC.ballistic=0.5;
generalFactors.fC.winged=0.5;

%% Cost conversion value
costConv=-7.905*(1e-6)*year^2+37.308156*(1e-3)*year-42.784828;

%% Cost calculation
%Development cost
CostDev=DevCost(engineEmptyWeight,generalFactors, costConv,nRocket,...
    nJet,jetEngineEmptyWeight,TotalLaunchMass,VehicleDryMass, wing,...
    rocketEngineCost,jetEngineCost);%

%Production cost
tempProdCost = ProdCost( engineEmptyWeight,generalFactors,costConv,...
    nRocket,nJet,jetEngineEmptyWeight,TotalLaunchMass,VehicleDryMass,...
    stageNumber, wing, rocketEngineCost);
ProdCosta=tempProdCost{1}+jetEngineCost.prod;
ProdCostVector=tempProdCost{2};
ProdCostVector.C_VJ=jetEngineCost.prod;

%Operating cost including propellant
OpeCosta = OpeCost(costConv,...
    TotalLaunchMass,VehicleDryMass,...
    TOmode,nLaunch,programLength,rates,numbUnits,CostDev,...
```

```
    ProdCosta,ProdCostVector,reuse,nPilots,nPAX,Tmiss, mProp, O_F,...
    Propellant, year)+jetEngineCost.maint+jetEngineCost.propellant;

%Cost of the additional vehicle
if (TOmode==0)
    additionalFixedCost=((-2e-8)*TotalLaunchMass^2+...
        0.0054*TotalLaunchMass+6.1159)*1e6*inflation(2015,year)*1.1;
    additionalVariableCost=((-3e-7)*TotalLaunchMass^2+0.1087*...
        TotalLaunchMass+1621.6)*inflation(2015,year)*1.1*1.5;
else
    additionalFixedCost=0;
    additionalVariableCost=0;
end

%Cost over the entire program
TotalCost=CostDev+additionalFixedCost+ProdCosta*numbUnits+...
    programLength*nLaunch*(OpeCosta+additionalVariableCost);

output=TotalCost;
end
```

```
function CostDev = DevCost( engineEmptyWeight,generalFactors,costConv,...
    nRocket, nJet, jetEngineEmptyWeight, Mtot, Mempty, wing,...
    rocketEngineCost, jetEngineCost)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Cost module: function that calculates the airframe development cost
% Author: Christopher Frank
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%Define variables
Mg=1e3; %Mass factor for cost calculation in kg
Mdollars=1e6; %Cost for price in $
fact=generalFactors;

%Airframe type
if wing==0
    C_DB=(42* Mtot/Mg + 30000)* Mdollars * fact.fC.ballistic*costConv;
    C_DW=0;

else
    C_DW=11350*Mdollars*((Mempty-nRocket*engineEmptyWeight-nJet*...
        jetEngineEmptyWeight)/Mg)^0.335 * fact.fPS * fact.fTD *...
        fact.fTQ.winged * fact.fTE * fact.fC.winged * costConv;
    C_DB=0;

end
```

```
26
27 %Compute body costs
28 C_Dbody=C_DW+C_DB;
29
30 %Compute jet engine cost
31 C_DJ=jetEngineCost.dev;
32
33 %Output
34 CostDev=C_Dbody+rocketEngineCost.DevCost+nJet*C_DJ;
35
36 end
```

```
1  function output=inflation(y1, y2)
2  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
3  % Cost module: inflation calculation
4  % Author: Christopher Frank
5  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
6
7  ir1=0.4627*y1-907.99;
8  ir2=0.4627*y2-907.99;
9
10 output=ir2/ir1;
11
12 end
```

```
1  function output = JetEngineCost(TIT, Wdry, LC, afterburner, year,...
2      tmission, nJet, Tj, fuelWeight)
3  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
4  % Cost module: function that computes all costs related to jet engines
5  % Author: Christopher Frank
6  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
7
8  %Conversions
9  drywt=Wdry*2.20462; %Dry weight (From kg to lbs)
10 TIT=(TIT-273.15)*1.8; %Turbine inlet temperature (From K to F)
11 Mdollars=1e6;
12
13 %Intermediate variable calculation
14 lnritf=log(TIT);
15 ab=afterburner;
16 lndrywt=log(drywt);
17 lnslope=log(LC*0.01);
```

```
18
19 %Development cost
20 lnrd01m=-24.429+4.027*lnritf ;
21 devCost=Mdollars*exp(lnrd01m)*1;%Replace 1 by cost conversion
22
23 %Production cost
24 lnT1 = -10.40 - 8.550*lnslope + 0.482*ab + 1.162*lnritf + 0.261*lndrywt;
25 prodCost=exp(lnT1)*Mdollars*1; %Replace 1 by cost conversion
26
27 %Maintenance cost
28 tFlight=tmission/3600; %Flight time (hr)
29 T0=Tj*0.224808943;
30 maintCost=tFlight*nJet*(75*(0.645+0.05*T0*1e-4)*(0.566+0.434/tFlight)...
31     +(25+0.05*T0*1e-4)*(0.62+0.38/tFlight));
32
33 %Outputs
34 output.dev=devCost*inflation(1981,year);
35 output.prod=prodCost*nJet*inflation(1981,year);
36 output.maint=maintCost*inflation(1995,year);
37 output.propellant=0.978*fuelWeight;
38
39 end
```

```
1  function OpeCost = OpeCost( costConv, Mtot,Mempty,TOmode,nLaunch,...
2      programLength,rates,numbUnits,CostDev,ProdCost,ProdCostVector,...
3      reuse, nPilots, nPAX, Tmiss, mProp, O_F, Propellant, year)
4  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
5  % Cost module: calculation of all operating cost components
6  % Author: Christopher Frank
7  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
8
9  Mg=1e3; %Mass factor for cost calculation in kg
10 Mdollars=1e6; %Cost for price in $
11 Thour=3600; %Time conversion for mission time in s
12
13 %% Variable Direct Operating Cost
14 %Pre-launch operating + Launch operating + Maintenance cost
15 switch TOmode
16     case 0
17         C_pl=1.409*Mdollars*nLaunch^-0.54*(Mtot/Mg)^0.75*costConv;
18         Cl=Mdollars*(14*nLaunch^-0.65 + 0.42*(Tmiss/Thour)*...
19             (nPilots+nPAX)^0.5*nLaunch^-0.8)*costConv;
20         Cmaint= 0.0001*ProdCostVector.C_Vbody+0.0015*...
21             ProdCostVector.C_VR+0.0002*ProdCostVector.C_VJ;
22
23     case 1
```

```
        C_pl=1.409*Mdollars*nLaunch^-0.54*(Mtot/Mg)^0.75*costConv;
        Cl=Mdollars*(20*nLaunch^-0.65 + 0.42*(Tmiss/Thour)*...
            (nPilots+nPAX)^0.5*nLaunch^-0.8)*costConv;
        Cmaint= 0.004*ProdCostVector.C_Vbody+0.0015*ProdCostVector.C_VR+...
            0.0002*ProdCostVector.C_VJ;

    case 2
        C_pl=6.618*Mdollars*nLaunch^-0.61*(Mtot/Mg)^0.75*costConv;
        Cl=Mdollars*(20*nLaunch^-0.65 + 0.42*(Tmiss/Thour)*...
            (nPilots+nPAX)^0.5*nLaunch^-0.8)*costConv;
        Cmaint= 0.004*ProdCostVector.C_Vbody+0.0015*ProdCostVector.C_VR...
            +0.0002*ProdCostVector.C_VJ;
end

%Propellant cost
SolidC=1110;
O2C=1761;
H2C=30764;
RP1C=21755;
N2O4C=226000;
MMHC=185120;
N2OC=2773;

if Propellant == 1
    propPrice=SolidC*mProp;
elseif Propellant ==2
    propPrice=O2C*mProp*O_F/(1+O_F)+H2C*mProp/(1+O_F);
elseif Propellant ==3
    propPrice=O2C*mProp*O_F/(1+O_F)+RP1C*mProp/(1+O_F);
elseif Propellant ==4
    propPrice=N2O4C*mProp*O_F/(1+O_F)+MMHC*mProp/(1+O_F);
elseif ((Propellant == 5) || (Propellant == 6))
    propPrice=O2C*mProp*O_F/(1+O_F)+SolidC*mProp/(1+O_F);
else
    propPrice=N2OC*mProp*O_F/(1+O_F)+SolidC*mProp/(1+O_F);
end
Cp=propPrice*inflation(2015,year)*1e-3;

%Launch site cost
Cls=0.1*Mdollars*costConv;

%Vehicle amortization cost
Cva=ProdCostVector.C_Vbody/reuse.Body + ProdCostVector.C_VJ/reuse.Jet +...
    ProdCostVector.C_VR/reuse.Rocket;

%Transportation cost
Ct=(72.1e-3)*Mdollars*((Mempty/Mg)^0.5)*costConv;

%Sum all variable DOC components
```

```
DOCv=C_pl+Cl+Cp+Cls+Cva+Ct+Cmaint;

%% Fixed Direct Operating Cost
%Development amortization cost
Cda=DOCv/(programLength*nLaunch);

%Financing cost
Cf=ProdCost * ((1+rates.IR)^rates.tIR * rates.RR*rates.tRR/...
    (1-(1+rates.RR)^(-rates.tRR)) -1)  *(1/(programLength*nLaunch));

%Product improvement cost
Cpi=0.045*((numbUnits*programLength)^0.33)*CostDev/...
    ((nLaunch*programLength));

%Abolition cost
Cab=numbUnits*DOCv/(nLaunch*programLength);

%Sum all cost fix DOC components
DOCf=Cda+Cf+Cpi+Cab;

%% Indirect Operating Cost (Administration cost per launch)
IOCtot=0.9*Mdollars*costConv;
%% Total operating cost without insurance
OpeCostNoIns=DOCv+DOCf+IOCtot; %$/launch

%% Vehicle cost with insurance
OpeCost=(OpeCostNoIns+rates.Rloss*ProdCost)/(1-rates.Rabort);

end
```

```
function output = ProdCost( engineEmptyWeight,generalFactors,costConv,...
    nRocket, nJet, jetEngineEmptyWeight, Mtot, Mempty, stageNumber,...
    wing, rocketEngineCost)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Cost module: function that calculates the production cost components
% Author: Christopher Frank
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

Mg=1e3; %Mass factor for cost calculation in kg
Mdollars=1e6; %Cost for price in $
fact=generalFactors;

%Airframe type
if wing==0
    C_VB=638.6*Mdollars*((Mempty-nRocket*engineEmptyWeight-nJet...
        *jetEngineEmptyWeight)/Mg)^0.485...
```

```
        *fact.fI^stageNumber * fact.fC.ballistic*fact.fCR.ballistic*...
        costConv;
    C_VW=0;

else
    C_VW=84.3*Mdollars*(Mtot/Mg)^0.669 *fact.fI^stageNumber * ...
        fact.fC.winged*fact.fCR.winged*costConv;
    C_VB=0;

end

%Compute body costs
C_Vbody=C_VW+C_VB;

%Outputs
ProdCostVector.C_VR=nRocket*rocketEngineCost.FUC;
ProdCostVector.C_Vbody=C_Vbody;
ProdCost=nRocket*rocketEngineCost.FUC+C_Vbody;

output={ProdCost, ProdCostVector};

end
```

```
function output=rocketCostModule(Q, Nn, LC, Proto,...
    rocketEngineEmptyWeight, Isp, propellantWeight, year, Propellant,...
    O_F, Tr, SOtank, SPtank)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Cost module: calculation of the costs related to rocket engines
% Author: Christopher Frank
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%% Intermediate calculation
g0=9.81;
coefLC=log(LC*0.01)/log(2);%Learning curve coefficient
mProp=propellantWeight*2.20462; %Propellant mass (lb)
T=Tr*0.224808943; %Thrust (lb)
mdot=T/Isp; %Mass flow (lb/sec)
Wt=rocketEngineEmptyWeight*2.20462;

%% Cost calculation
if Propellant == 1
    %Total impulse calculation
    TI=Isp*mProp*0.001; %Total impulse (klb.sec)

    %Estimation based on Total Impulse
    CAC_TI=1e3*77.595*(Q^(-0.3597)) * TI^0.5081 * Nn^0.6116;
```

```
24
25      %Calculation of First Unit Cost
26      FUC=CAC_TI/Q^coefLC;
27
28      %Calculation of the development cost
29      CAC_150=inflation(1988,1987)*FUC*150^coefLC;
30      DevCost=52.947*CAC_150^0.939*Proto^0.618;
31
32      %Ouputs assignment
33      FUC=inflation(1988,year)*FUC;
34      DevCost=inflation(1987,year)*DevCost;
35
36  elseif (Propellant <4.5)
37      %Estimation of FUC
38      FUC=374 * Wt^1.718*(T^(-0.043))*(mdot^(-0.827));
39
40      %Calculation of the development cost
41      CAC_150=inflation(1965,1987)*FUC*150^coefLC;
42      DevCost=52.947*CAC_150^0.939*Proto^0.618;
43
44      %Ouputs assignment
45      FUC=inflation(1965,year)*FUC;
46      DevCost=inflation(1987,year)*DevCost;
47
48  else
49      %Solid sub-engine
50      WsolProp=propellantWeight/(1+O_F); %Weight of solid propellant (kg)
51      Wsol=2.20462*(0.0706*WsolProp+18.974); %Solid sub-engine weight (lbs)
52      CAC_Q=1000*29.045*Q^(-0.3387)*(Wsol+2.20462*WsolProp)^(0.5126)...
53          *Nn^(0.6167);
54      solidSubCost88=CAC_Q/Q^coefLC; %Calculation of First Unit Cost %1988
55
56      %Tanks
57      Ctr=328*(SOtank+SPtank); %1975
58      Cnr=2660*(SOtank+SPtank);
59
60      %Feed system
61      %Liquid sub-engine weight(lbs)
62      Wliq=2.20462*Tr/(g0*(25.2*log10(Tr)-80.7));
63      feedCost=398.2*Wliq^(0.618); %1965
64
65      %Development cost
66      %Solid sub-engine
67      CAC_150Solid=inflation(1988,1987)*solidSubCost88*150^coefLC;
68      CAC_150Tank=inflation(1975,1987)*(Ctr+Cnr); %CAC150 tanks
69      CAC_150Feed=inflation(1965,1987)*feedCost;
70      DevCost=52.947*(CAC_150Feed+CAC_150Tank+CAC_150Solid)^0.939...
71          *Proto^0.618;
72
```

```
    %Ouputs assignment
    FUC=(CAC_150Feed+CAC_150Tank+CAC_150Solid)*inflation(1987,year)*1.1;
    DevCost=DevCost*inflation(1987,year);
end

%Outputs
output.FUC=FUC;
output.DevCost=DevCost;
end
```

### D.2.4 Propulsion Module

```
function output= gammaLiq(O_F, Propellant)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Propulsion module: Gamma calculation for each type of liquid propellant
% Authors: Christopher Frank and Clemence Tyl
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

switch Propellant
    case 'LOXLH2'
        a1 =       1.617  ;
        b1 =      -7.825  ;
        c1 =       6.251  ;
        a2 =         1.2  ;
        b2 =       9.087  ;
        c2 =       38.53  ;
        gamma =  a1*exp(-((O_F-b1)/c1)^2) + a2*exp(-((O_F-b2)/c2)^2);

    case 'LOXRP1'
        a =      0.9455  ;
        b =      -1.753  ;
        c =       1.204  ;
        d =    0.001297  ;
        gamma = a*exp(b*O_F) + c*exp(d*O_F);

    case 'Hypergolic'
        p1 =  -0.0005371  ;
        p2 =    0.008974  ;
        p3 =    -0.05822  ;
        p4 =       0.183  ;
        p5 =     -0.2685  ;
        p6 =       1.372  ;
        gamma = p1*O_F^5 + p2*O_F^4 + p3*O_F^3 + p4*O_F^2 + p5*O_F + p6;
end

%Output
```

```
output=gamma;

end
```

```
function output = JetEngines( Tj, BPR, MmaxJet, afterburner, TIT )
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Propulsion module: weight, size, and performance calculation of jet
% engines
% Author: Christopher Frank
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% Calculate the impact of TIT
Tj=Tj*0.001; %Conversion of thrust from N to kN
deltaTIT=(TIT-1500)/1500;
deltaSFC=-0.511346662342284 + 24.34814880625 * deltaTIT +...
    0.29773126105 * BPR + (deltaTIT - 0.233333333) * ...
    (BPR - 17.5) * 0.8306873895;
eta=1-(deltaSFC-2.88062683993704)*0.01;

% Calculate required parameters for each type of engine
switch afterburner
    case false
        output.weight=14.7*Tj^1.1*exp(-0.045*BPR);
        output.length=0.49*Tj^0.4*MmaxJet^0.2;
        output.diameter=0.15*Tj^0.5*exp(0.04*BPR);
        output.SFC=eta*19*exp(-0.12*BPR)*1e-6;
    case true
        output.weight=11.1*Tj^1.1*MmaxJet^0.25*exp(-0.81*BPR);
        output.length=0.68*Tj^0.4*MmaxJet^0.2;
        output.diameter=0.11*Tj^0.5*exp(0.04*BPR);
        output.SFC=eta*60*exp(-0.12*BPR)*1e-6;
end

end
```

```
function output = massHybrid(mprop,time,epsilon,pc,D, cc, O_F, matMC,...
    Propellant)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Propulsion module: weight/size calculation for hybrid rocket engines
% Authors: Christopher Frank and Clemence Tyl
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

```

```
%% Constants
g0=9.81;
N=8;% Assuming a wagon wheel with no center hole burning, 8 triangle ports
Gox_i=350;%kg/m^2.s Assumption: data from Humble
pc=pc*1e6;%Conversion: MPa -> Pa
materialTankOx=matMC;

%% Provide densities to fuel and oxidizer according to the selected
% propellant
switch Propellant
    case 5
        a=9.26*10^(-6); %AIAA 2012-4202
        n=0.852;
        m=0;
        rho_f=1142;
        rho_ox=930;
    case 6
        a=9.1*10^(-5); %AIAA 2012-4202
        n=0.69;
        m=0;
        rho_f=1142;
        rho_ox=900;
    case 7
        a=1.87*10^(-4);%AIAA 2012-4202
        n=0.347;
        m=0;
        rho_f= 1223;
        rho_ox=930;
    case 8
        a=1.315*10^(-4);%AIAA 2011-420 12
        n=0.555;
        m=0;
        rho_f= 1223;
        rho_ox=900;
end

db=mprop/time;%propellant mass flow rate kg/s
mfuel=mprop/(O_F+1);%kg
mox=mprop/(1/O_F+1);%kg
rho_matOx=materialTankOx(1);%kg/m^3
Ftu_Ox=materialTankOx(2);%Pa

%% Solid part
%Configuration of the combustion ports
dbf=mfuel/time;
dbox=mox/time;

%Initial geometry of the ports
Api=dbox/(N*Gox_i);
```

```
Gf_i=Gox_i/O_F;
theta_p=pi/N;
hi=sqrt(Api/tan(theta_p));
bi=2*hi*tan(theta_p);
li=hi/cos(theta_p);
P_i=2*hi/cos(theta_p)+bi;

Lp=(dbf/(N*rho_f*a*(Gox_i+Gf_i)^n*P_i))^(1/(1+m));%port length

%Final configuration
aaa=1;
bbb=(2*li+bi)/pi;
ccc=-mfuel/(N*rho_f*Lp*pi);
delta=bbb^2-4*aaa*ccc;

w=(-bbb+sqrt(delta))/(2*aaa);

%Radius of central hole
rh=w/sin(theta_p)-w;

%Radius of grain
rg=hi+2*w+rh;

%Combustion chamber size and weight
pb_chamber=1.5*pc;

Lc=Lp+2*rg;%combustion chamber length
tc=pb_chamber*rg*1.5/Ftu_Ox;%wall thickness
mcase=pi*2*rg*Lc*tc*rho_matOx;%chamber case mass

%% Injector mass
%2.5cm thick aluminum plate spanning the chamber radius humble p438
minj=2800*pi*rg^2*0.025;

%% Nozzle
At=db*cc/pc;
Ae=epsilon*At;
De=sqrt(4*Ae/pi);
Dt=sqrt(4*At/pi);
Lnozzle=(De-Dt)/(2*tan(15*pi/180))*0.8;%15 half angle, 0.8 bell nozzle
mnozzle=125*(mprop/5400)^(2/3)*(epsilon/10)^(1/4);

%% Liquid part
%Oxidizer tank
%Volume with 10% ullage
Vox=1.1*(mox/rho_ox);
pbox=pc+0.2*pc+50000+0.5*rho_ox*10^2;
Rsox=(3*Vox/(4*pi))^(1/3);
if Rsox>D/2
```

```
106     %cylindrical part
107     rcox=D/2;
108     lcox=(Vox-4/3*pi*rcox^3)/(pi*rcox^2);
109     Acox=2*pi*rcox*lcox;
110     tcox=(pbox*rcox)/Ftu_Ox;
111     mcox=Acox*tcox*rho_matOx;
112     %spherical part
113     Asox=4*pi*rcox^2;
114     tsox=(pbox*rcox)/(2*Ftu_Ox);
115     msox=Asox*tsox*rho_matOx;
116
117     mTankOx=mcox+msox;
118     Loxtank=lcox+D;
119     SOtank=Acox+Asox;
120 else
121     %just a sphere
122     lsox=2*Rsox;
123     Asox=4*pi*Rsox^2;
124     tsox=(pbox*Rsox)/(2*Ftu_Ox);
125     msox=Asox*tsox*rho_matOx;
126
127     mTankOx=msox;
128     Loxtank=lsox;
129     SOtank=Asox;
130 end
131
132 %Pressurization tank: Helium
133 rHe=8314/4.003;%J/kg/K
134 gammaHe=1.66;
135 p_f=pbox;
136 p_i=21*10^6;
137 Ti=273;
138 Tf=Ti*(p_f/p_i)^((gammaHe-1)/gammaHe);
139
140 %Loop
141 VHe=0;
142 Vtank=Vox+VHe;
143 mHe=(1.05*p_f*(Vtank))/(rHe*Tf);
144 VpHe=(mHe*Ti*rHe)/p_i;
145
146 while (abs(VpHe-VHe)>=0.0001)
147     VHe=VpHe;
148     Vtank=Vox+VHe;
149     mHe=(1.05*p_f*(Vtank))/(rHe*Tf);
150     VpHe=(mHe*Ti*rHe)/p_i;
151 end
152
153 VHe=VpHe;
154
```

```
%Helium tank dimensions
RsHe=(3*VHe/(4*pi))^(1/3);
if RsHe>D/2
    %cylindrical part
    rcHe=D/2;
    lcHe=(VHe-4/3*pi*rcHe^3)/(pi*rcHe^2);
    %AcHe=2*pi*rcHe*lcHe;
    %spherical part
    %AsHe=4*pi*rcHe^2;
    LHeTank=lcHe+D;
    SPtank=2*pi*rcHe*lcHe+pi*RsHe^2;
else
    %just a sphere
    lsHe=2*RsHe;
    %AsHe=4*pi*RsHe^2;
    LHeTank=lsHe;
    SPtank=pi*RsHe^2;
end

%Helium tank mass
%(pV/W approach)
MTankPress=p_i*VHe/(g0*12700);%12700 p439  or 6350 titanium p279 humble

%% Support structure and ancillary parts %%%%%%%%
mStructure=0.1*(mTankOx+MTankPress);

%% Total
mtot_empty=mcase+minj+mnozzle+mTankOx+MTankPress+mStructure+mHe;
Ltot=Lc+Lnozzle+Loxtank+LHeTank;

%% Ouputs
output.length=Ltot;
output.mEmpty=mtot_empty+mHe;
output.SOtank=SOtank;
output.SPtank=SPtank;

end
```

```
function output = massLiquid(mProp, tc, materialMC, DD, Isp, O_F,...
    Propellant)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Propulsion module: weight/size calculation for liquid rocket engines
% Authors: Christopher Frank and Clemence Tyl
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%% Constants
```

```
g0=9.81;
MEOP=1.2;%pc_max/pc
fs=2;%typically 2 for pressure vessels

%% Provide densities to fuel and oxidizer according to the selected
% propellant
switch Propellant
    case 2
        rho_ox=1142;
        rho_fuel=71;
    case 3
        rho_ox=1142;
        rho_fuel=810;
    case 4
        rho_ox=1440;
        rho_fuel=878;
end

db=mProp/tc;%kg/s
F=Isp*db*g0;%N

D=DD-0.1;
mfuel=mProp/(O_F+1);%kg
mox=mProp/(1/O_F+1);%kg

rho_mat=materialMC(1);%kg/m^3
Ftu=materialMC(2);%Pa

%% First estimation, mass and dimensions %%%%
mE=F/(g0*(25.2*log10(F)-80.7)); %kg
LE=(0.00003042*F+327.7)*10^(-2);%m

%% Tanks volume and mass
Vf=1.1*(mfuel/rho_fuel);%m^3
Vox=1.1*(mox/rho_ox);%m^3

%% Tank pressure and burst pressure
ptf=(10^(-0.1068*(log10(Vf)-0.2588)))*10^6;
ptox=(10^(-0.1068*(log10(Vox)-0.2588)))*10^6;
pbf=MEOP*ptf*fs;
pbox=MEOP*ptox*fs;

%% Fuel tank dimensions
Rsf=(3*Vf/(4*pi))^(1/3);
if Rsf>D/2
    %%%%%%%%%%%cylindrical part
    rcf=D/2;
    lcf=(Vf-4/3*pi*rcf^3)/(pi*rcf^2);
    Acf=2*pi*rcf*lcf;
```

```
58      tcf=(pbf*rcf)/Ftu;
59      mcf=Acf*tcf*rho_mat;
60      %%%%%%%%%%%%%%spherical part
61      Asf=4*pi*rcf^2;
62      tsf=(pbf*rcf)/(2*Ftu);
63      msf=Asf*tsf*rho_mat;
64
65      mftank=mcf+msf;
66      Lftank=lcf+D;
67
68  else
69      %Just a sphere
70      lsf=2*Rsf;
71      Asf=4*pi*Rsf^2;
72      tsf=(pbf*Rsf)/(2*Ftu);
73      msf=Asf*tsf*rho_mat;
74
75      mftank=msf;
76      Lftank=lsf;
77  end
78
79  %% Oxidizer tank dimensions
80  %Cylindrical part
81  Rsox=(3*Vox/(4*pi))^(1/3);
82  if Rsox>D/2
83      rcox=D/2;
84      lcox=(Vox-4/3*pi*rcox^3)/(pi*rcox^2);
85      Acox=2*pi*rcox*lcox;
86      tcox=(pbox*rcox)/Ftu;
87      mcox=Acox*tcox*rho_mat;
88      %Spherical part
89      Asox=4*pi*rcox^2;
90      tsox=(pbox*rcox)/(2*Ftu);
91      msox=Asox*tsox*rho_mat;
92
93      moxtank=mcox+msox;
94      Loxtank=lcox+D;
95
96  else
97      %Just a sphere
98      lsox=2*Rsox;
99      Asox=4*pi*Rsox^2;
100     tsox=(pbox*Rsox)/(2*Ftu);
101     msox=Asox*tsox*rho_mat;
102
103     moxtank=msox;
104     Loxtank=lsox;
105 end
106
```

```
%% Mass attachements
m_attachement=0.453592*1.949e-3 * (F*0.224808943)^1.0687;

%% Case in aluminum, 6mm thickness
mcase=2*pi*D/2*(Lftank+Loxtank)*0.006*2800;

%% Total mass
mtot=mE+mftank+moxtank+m_attachement+mcase;
L_int_E=LE+Lftank+Loxtank;

%% Outputs
output.length=L_int_E;
output.mEmpty=mtot;

end
```

```
function output = massSolid(mProp, tc, epsilon, thrust, pc, Isp)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Propulsion module: weight/size calculation for solid rocket engines
% Author: Christopher Frank
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%Implementation of the surrogate models calculated in JMP
emptyWeight= 1002.02283765734 + 0.0967425683356674 .* mProp ...
    -149.552932619832 .* pc + (pc - 6.8321052631579) .* ...
    (mProp - 7857.26578947369) .* (-0.0262668618042834);

diameter =0.443293470488469 + 0.0000077467641535731 .* mProp + ...
    0.0121151381460204 .* tc + 0.0250534295480696 .* pc + ...
    0.00945529003061431 .* epsilon + -0.00260675602417801 .*...
    Isp + 0.000000444291436658 .* thrust;

length=25.0394552567526 + 0.000503146858282769 .* mProp ...
    -0.0770093558319263 .* Isp + (mProp - 11333.8071428571) .* ...
    (mProp - 11333.8071428571) .* -0.000000014318090183 + ( tc -...
    63.8285714285714).*(tc - 63.8285714285714) * -0.000800697313856403;

%Outputs
output.mEmpty=emptyWeight;
output.length=length;
output.diameter=diameter;

end
```

```
function output = performanceCalculation(pc, epsilon, Propellant)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Propulsion module: performance calculation for rocket engines
% Author: Christopher Frank
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%Calibration parameters
calibPerfo.Solid=0.9272;
calibPerfo.LOXLH2=0.9565938;
calibPerfo.LOXRP1=0.9166761;
calibPerfo.Hypergolic= 0.9248941;
calibPerfo.LOXHTPB= 0.7728613;
calibPerfo.LOXParaffin= 0.7649672;
calibPerfo.N2OHTPB= 0.8256039;
calibPerfo.N2OParaffin= 0.7899722;

%Calculate performance parameters for each propellant
switch Propellant
    case 1
        Isp = calibPerfo.Solid .* (255.427396710744 + 0.197108393221772...
            .*log( pc ) + 15.5355097527882 .*log( epsilon ));
        cc = calibPerfo.Solid .* ( 1569.52772832855 + 9.33887410822712 ...
            .* log( pc )  -0.00373725577675696 .* log( epsilon ));
        OF=[];

    case 2
        Isp = calibPerfo.LOXLH2.* (388.512079641684 + ...
            0.602021396435942 .* log( pc ) + 17.0859008700137 .*...
            log( epsilon ));
        cc = calibPerfo.LOXLH2.* (2460.57803850249 +...
            6.39478573234992 .* log( pc )  -22.6812882052718 .*...
            log( epsilon ));
        OF=2.85978023285533 + 0.0813655032843058 * log( pc ) + ...
            0.435726055292276 * log( epsilon );

    case 3
        Isp= calibPerfo.LOXRP1.*(283.719601145181 + ...
            1.75816886543178 .* log( pc ) + 18.5944818419438 ...
            .* log( epsilon ));
        cc = calibPerfo.LOXRP1.*(1784.904265689 + 17.1274817632888...
            .* log( pc )  -9.62724016370026 .* log( epsilon ));
        OF=2.20012405155977 + 0.0455192073177058 * log( pc ) + ...
            0.115424459015865 * log( epsilon );

    case 4
        Isp= calibPerfo.Hypergolic.*(279.412912491884 +...
            0.910668022569868 .* log( pc )+15.6798582616569.*log(epsilon));
        cc= calibPerfo.Hypergolic.*(1747.24422351986 + ...
            7.5107035219132 .* log(pc) -10.7171560187922 .* log( epsilon));
```

```
        OF=1.51686551440224 + 0.0683768358725008 * log( pc ) +...
            0.145375368172541 * log( epsilon );

    case 5
        Isp = calibPerfo.LOXHTPB.*(277.331650951675 + 2.16558350827057...
            .*log( pc ) +  18.4878815011429 .*log( epsilon ));
        cc = calibPerfo.LOXHTPB.*(1751.92905036163 + 17.3998450267533 ...
            .* log( pc ) -9.44640819512984 .* log( epsilon ));
        OF= 1.93293875106783 + 0.0474765699066492 * log( pc ) +...
            0.109562010700826 * log( epsilon );

    case 6
        Isp = calibPerfo.LOXParaffin.*(283.024062942826 + ...
            2.33441220682802 .*log(pc)+18.5884665855212 .*log( epsilon ));
        cc = calibPerfo.LOXParaffin.*(1787.96670366732 +16.7203151778903...
            .*log( pc )  -9.54373487317511 .*log( epsilon ));
        OF = 2.24358662271426 + 0.0485752731817214 * log( pc ) +...
            0.115354995865352 * log( epsilon );

    case 7
        Isp = calibPerfo.N2OHTPB.*(251.959800040605 + 1.09012877785973...
            .*log( pc ) + 13.5705733933019 .*log( epsilon ));
        cc = calibPerfo.N2OHTPB.*(1569.51888535442 + 12.1271710330801...
            .* log( pc )  -8.03074535000083 .* log( epsilon ));
        OF= 6.21730103691382 + 0.0850479578806618 * log( pc ) + ...
            0.447075852597881 * log( epsilon );

    case 8
        Isp = calibPerfo.N2OParaffin.*(255.139146758501 + ...
            0.497931499747107.*log( pc )+13.2784633299985.*log( epsilon ));
        cc = calibPerfo.N2OParaffin.*(1576.10666690323 +12.0223666953543...
            .* log( pc )  -7.10683369810811 .* log( epsilon ));
        OF= 7.31981068298658 + 0.0523080610999142 * log( pc ) +...
            0.417750655765745 * log( epsilon );
end

%Outputs
output.Isp=Isp;
output.cc=cc;
output.O_F=OF;

end
```

```
function output = propuTrajectory(h, perfoRocket, epsilon, pc)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Propulsion module: determine the Isp required for the trajectory
```

```
% optimization module
% Author: Christopher Frank
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%Load information
cc=perfoRocket.cc;
Ispv=perfoRocket.Isp;
pcPa=pc*1e6; %From MPa to Pa
atm=funcAtm(h);

%Output
Isp=Ispv-epsilon*cc*atm.p/(pcPa*atm.g);
output=Isp;

end
```

```
function output=RocketEngines(pc, epsilon, mProp, combTime, thrust,...
    engineDiameter, Propellant)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Propulsion module: function that drives the performance and weight/size
% calculation of rocket engines
% Author: Christopher Frank
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%Material definition
matMC=[2800 0.413*1e9;%Aluminum
    4460 1.23*1e9;%Titanium
    7830 1.52*1e9;%D6aC Steel
    7830 0.862*1e9;%4130 Steel
    1550 0.5*(0.965+1.72)*1e9;%Graphite
    1380 0.5*(0.827+1.1)*1e9;%Kevlar
    1990 1.1e9;%Fiberglass
    ];
materialMC=[matMC(1,1) matMC(1,2)]; %Need to pick one !

%Call performance calculation function
perfo = performanceCalculation(pc, epsilon, Propellant);

%Call weight/size calculation function
weightSize = weightSizeCalculation(pc, epsilon, Propellant, ...
    thrust, mProp, combTime, perfo.Isp, perfo.cc, perfo.O_F,...
    materialMC, engineDiameter);

%Outputs
output.weightEngine=weightSize.emptyWeight;
output.lengthEngine=weightSize.length;
```

```
output.rocketPerfo=perfo;
output.SOtank=weightSize.SOtank;
output.SPtank=weightSize.SPtank;

end
```

```
function output = weightSizeCalculation(pc, epsilon, Propellant,...
    thrust, mProp, tc, Isp, cc, O_F, materialMC, D)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Propulsion module: determine the weight and size of the rocket engine
% Author: Christopher Frank
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

if Propellant==1 %Solid engine
    massSize=massSolid(mProp, tc, epsilon, thrust, pc, Isp);
    output.SOtank= [];
    output.SPtank= [];


elseif  ((Propellant==2) || (Propellant==3) || (Propellant==4))
    %Liquid engine
    massSize=massLiquid(mProp, tc, materialMC, D, Isp, O_F, Propellant);
    output.SOtank= [];
    output.SPtank= [];

elseif ((Propellant==5) || (Propellant==6) || (Propellant==7) ||...
        (Propellant==8)) %Hybrid engine
    massSize=massHybrid(mProp, tc, epsilon, pc, D, cc, O_F, materialMC,...
        Propellant);
    output.SOtank= massSize.SOtank;
    output.SPtank= massSize.SPtank;

else %Wrong engine type
    fprintf('Please enter a valid engine type')
end

%Outputs
output.emptyWeight=massSize.mEmpty;
output.length=massSize.length;

end
```

### D.2.5 Trajectory Module

```
function output = climbJet(Tj, nJet, Wchanging, aeroCd0, Swing, h,...
    Dh, sweep, AR, TR, tc, Vopt)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Trajectory module: calculate the optimum configuration that corresponds
% to the maximum rate of climb
% Author: Christopher Frank
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%Constants and parameters definition
g0=9.81;
rho0=1.225;
atm=funcAtm(h);
rho=atm.rho;
Tact=Tj*rho/rho0;
M=VtoM(Vopt,h);

%Aerodynamic model
coef=aeroCd0.coef;
Cd1=aeroCd0.Cd1;
coefcoef=aeroCd0.coefcoef;
COEF=aeroCd0.COEF;
Cd06=aeroCd0.Cd06;
Cd12=aeroCd0.Cd12;
Cd0=aeroTrajectory(M, h, Cd06, Cd12, coef, Cd1, coefcoef, COEF);
spanEff=spanEfficiency(sweep, AR, TR, tc, M);
L_Dmax=0.5*sqrt(pi*AR*spanEff/Cd0);

%Thrust and weight model
T=Tact*nJet;
W=g0*Wchanging; %kg->N
Z=1+(1+ 3/((L_Dmax*T/W)^2) )^0.5;

%Maximum ROC calculation
ROCmax=(W*Z/(Swing*3*rho*Cd0))^0.5  *(T/W)^1.5  * ...
    (1- Z/6  -3/(2*Z*(L_Dmax*T/W)^2));

%Speed calculation
Vopt= (T/(Swing*3*rho*Cd0) * (1+ sqrt( 1+3/((L_Dmax*T/W)^2))))^0.5;

%Outputs
output.Dt=Dh/ROCmax; %Time spent (s)
output.V=Vopt;
output.Tact=T;

end
```

```
function [c , ceq]  = constraintsDefinition(x, E, aeroCd0, Tr, mass,...
    Sref, nMax)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Trajectory module: constraints definition
% Author: Christopher Frank
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%Definition of the constraints for the trajectory
qMax=50000;
tMax=2500;

%Variables definition
M=x(1);
h=x(2);
g=9.81;
V=MtoV(M,h);
ceq=E-0.5*V^2-g*h;
gamma=1.4;

%Drag definition
atm=funcAtm(h);
rho=atm.rho;
coef=aeroCd0.coef;
Cd1=aeroCd0.Cd1;
coefcoef=aeroCd0.coefcoef;
COEF=aeroCd0.COEF;
Cd06=aeroCd0.Cd06;
Cd12=aeroCd0.Cd12;
Cd=aeroTrajectory(M, h, Cd06, Cd12, coef, Cd1, coefcoef, COEF);
D=0.5*rho*Sref*(V^2)*Cd;

%Cosntraints definition
dynamicPressure=0.5*atm.rho*V^2-qMax;
load_factor=(Tr-D)/(mass*g)-0.5*nMax;
temp_max=atm.T+(gamma-1)*atm.T*M^2/gamma-tMax;

%Outputs
c=[dynamicPressure, load_factor, temp_max];

end
```

```
function output = LandingModule(sweepWing, Swing, VehicleDryMass)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Trajectory module: calculation of the take-off performance of the vehicle
% Author: Christopher Frank
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```

%Constants
rho0=1.2250;
g0=9.81;
muR=0.7;

%Landing performance
CLmaxLA=0.9*3*cos(sweepWing);
Vla=1.3*sqrt((VehicleDryMass*g0)/(0.5*Swing*rho0*CLmaxLA));
Lpr=(0.5*Vla^2)/(muR*g0);
LpLA=Lpr/0.6;

%Outputs
output.TOFL=LpLA;
output.landingSpeed=Vla;

end
```

```
function  optim  = optim(x, aeroCd0, perfoRocket, pc, epsilon, Tr, Sref)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Trajectory module: Definition of the objective function for the
% optimzation algorithm
% Author: Christopher Frank
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%Variable definition
M=x(1);
h=x(2);
atm=funcAtm(h);
rho=atm.rho;
V=MtoV(M,h);

%Performance definition
Isp=propuTrajectory(h, perfoRocket, epsilon, pc);
coef=aeroCd0.coef;
Cd1=aeroCd0.Cd1;
coefcoef=aeroCd0.coefcoef;
COEF=aeroCd0.COEF;
Cd06=aeroCd0.Cd06;
Cd12=aeroCd0.Cd12;

%Drag calculation
Cd=aeroTrajectory(M, h, Cd06, Cd12, coef, Cd1, coefcoef, COEF);
D=0.5*rho*Sref*(V^2)*Cd;

%Function to minimize
```

```
optim=((D-Tr)*V*atm.g*Isp)/Tr;
end
```

```
function output = TOModule(sweepWing, Swing, TotalLaunchMass, T07)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Trajectory module: calculation of the take-off performance of the vehicle
% Author: Christopher Frank
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%Constants
rho0=1.2250;
g0=9.81;

%Take-off performance
CLmaxTO=0.9*2.8*cos(sweepWing);
Vto=1.2*sqrt((TotalLaunchMass*g0)/(0.5*Swing*rho0*CLmaxTO));
Lpml=(TotalLaunchMass*g0)^2/(Swing*CLmaxTO*T07);
LpTO=0.17*Lpml+27;

%Ouputs
output.TOFL=LpTO;
output.TOSpeed=Vto;

end
```

```
function output = trajectoryJet(Tj, takeOffWeight, aeroCd0, Swing, nJet,...
    afterburner, TIT, BPR, hInitial, hTransition, sweep, AR, TR, tc,...
    LAmode, EmptyWeight)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Trajectory module: calculation of the optimum climb trajectory for jet
% engine climb
% Author: Christopher Frank
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%% Initialization
steps=250;
hVec=hInitial:(hTransition-hInitial)/steps:hTransition;
Dh=(hTransition-hInitial)/steps;

%TSFC calculation
deltaTIT=(TIT-1500)/1500;
deltaSFC=-0.511346662342284 + 24.34814880625 * deltaTIT +...
```

```
    0.29773126105 * BPR + (deltaTIT - 0.233333333) * ...
    (BPR - 17.5) * 0.8306873895;
eta=1-(deltaSFC-2.88062683993704)*0.01;

if (afterburner ==0)
    TSFC=eta*19*exp(-0.12*BPR)*1e-6; %kg/Ns
else
    TSFC=eta*60*exp(-0.12*BPR)*1e-6;
end

DwTot=0.03*takeOffWeight; %Warmup and take-off (Raymer p. 21)
Wchanging=takeOffWeight-DwTot;

%% Loop
Vopt=0;
for i=1:steps
    h=hVec(i);
    climbResults=climbJet(Tj, nJet, Wchanging, aeroCd0, Swing, h,...
        Dh, sweep, AR, TR, tc, Vopt);
    Dt=climbResults.Dt;
    Vopt=climbResults.V;
    Tact=climbResults.Tact;
    if Dt < 0 %Stop the loop when service ceiling is reached
        error('Service ceiling reached')
        break
    end
    %Calculate the variables needed
    Dw=TSFC*Dt*Tact*nJet;
    Wchanging=Wchanging-Dw;
    DwTot=DwTot+Dw;

end

%% Fuel consumption for horizontal powered landing
if LAmode == 1
    DwTot=DwTot+EmptyWeight*0.005/0.995;
end

%% Attribute outputs
output.WendClimbJet=Wchanging;
output.Dw=DwTot;
output.Vopt=Vopt;

end
```

```
function output = trajectoryModule(takeOffWeight, aeroCd0, perfoRocket,...
```

```
    pc, epsilon, Tr, Sref, nSteps, hMax, hMin, nMax)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Trajectory module: function that drives the trajectory optimization
% Author: Christopher Frank
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%% Optimization parameters
x0=[0,0]; %Initial conditions [Mach number, altitude (m)]
a=[]; %Linear inequality constraint matrix (a*x<=b)
b=[]; %Linear inequality constraint result (b*x<=b)
aeq=[]; %Linear equality constraint matrix (aeq*x=b)
beq=[]; %Linear equality constraint result (beq*x=b)
lb=[0,0]; %Lower limit for x variable
ub=[7, 100000]; %Upper limit for x variable (SI)
g0=9.81;


%% Initialization
EsMax=g0*hMax; %Maximum specific energy needed to reach hMax (m)
EsMin=g0*hMin; %Initial energy (m)
dtVec=[];
vecE=EsMin:(EsMax-EsMin)/nSteps:EsMax;
mass=takeOffWeight;

%% Optimization loop
for k=2:nSteps-1
    E=vecE(k);
    options=optimset('Display','off','TolX',0.00001,'TolFun',0.00001);
    [x, fval]=fmincon(@(x)optim(x, aeroCd0, perfoRocket, pc, epsilon,...
        Tr, Sref),x0,a,b,aeq,beq,lb,ub,@(x)constraintsDefinition(x,...
        E, aeroCd0, Tr, mass, Sref, nMax),options);

    Isp=propuTrajectory(x(2), perfoRocket, epsilon, pc);
    dm_m=-((EsMax-EsMin)/nSteps)/fval;
    dt=dm_m*mass*g0*Isp/Tr;
    dtVec= [dtVec dt];
    mass=mass*(1-dm_m);

end

%% Attribute output variables
output.totalTime=sum(dtVec);
output.propNeeded=takeOffWeight-mass;

end
```

### D.2.6 Safety Module

```
function output = safetyModule (pc, Propellant, LAmode, TOmode, nJet,...
    nPilots, JetEngine, Tr, Vla, Vto, afterburner)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Safety module: determine the risk level of the vehicle by decomposing the
% vehicle into functions
% Author: Christopher Frank
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%% Rocket propulsion severity factor
pcmin=2;
pcmax=8;
impactPc=0.05;
pcmean=0.5*(pcmin+pcmax);
deltaP=0.5*(pcmax-pcmin);

Trmin=100000;
Trmax=500000;
impactTr=0.03;
Trmean=0.5*(Trmin+Trmax);
deltaTr=0.5*(Trmax-Trmin);

switch Propellant

    case 1 % solid
        xRocketPropu= 8*(1+impactPc*((pc-pcmean)/deltaP))*...
            (1+impactTr*((Tr-Trmean)/deltaTr));

    case 2 % Lox/Lh2
        xRocketPropu= 5*(1+impactPc*((pc-pcmean)/deltaP))*...
            (1+impactTr*((Tr-Trmean)/deltaTr));

    case 3 % Lox/Rp1
        xRocketPropu= 4.5*(1+impactPc*((pc-pcmean)/deltaP))*...
            (1+impactTr*((Tr-Trmean)/deltaTr));

    case 4 % Hypergolic
        xRocketPropu= 6*(1+impactPc*((pc-pcmean)/deltaP))*...
            (1+impactTr*((Tr-Trmean)/deltaTr));

    case 5 % Lox/HTPB
        xRocketPropu= 2*(1+impactPc*((pc-pcmean)/deltaP))*...
            (1+impactTr*((Tr-Trmean)/deltaTr));

    case 6 % Lox/Paraffin
        xRocketPropu= 1*(1+impactPc*((pc-pcmean)/deltaP))*...
            (1+impactTr*((Tr-Trmean)/deltaTr));

    case 7 % N2O/HTPB
        xRocketPropu= 3*(1+impactPc*((pc-pcmean)/deltaP))*...
```

```
            (1+impactTr*((Tr-Trmean)/deltaTr));

    case 8 % N2O/ Parrafin
        xRocketPropu= 2*(1+impactPc*((pc-pcmean)/deltaP))*...
            (1+impactTr*((Tr-Trmean)/deltaTr));
end

%% Jet engines severity factor
JEngineMax=8;
JEngineMin=0;
impactJEngine=0.2;
JEnginemean=0.5*(JEngineMin+JEngineMax);
deltaJEngine=0.5*(JEngineMax-JEngineMin);
jetCoef=min(8-2*nJet,2);
if JetEngine==0
    xJetEngines=9;
else
    if afterburner==0
        xJetEngines=2*(1+impactJEngine*((jetCoef-JEnginemean)...
            /deltaJEngine));
    else
        xJetEngines=5*(1+impactJEngine*((jetCoef-JEnginemean)...
            /deltaJEngine));
    end
end

%% Pilots severity factor
xPilots=8-2*nPilots;

%% Launch severity factor
if TOmode == 0
    xTOmode=8;
elseif TOmode == 2
    xTOmode=4;
else
    Vtomin=50;
    Vtomax=200;
    impactVto=0.05;
    Vtomean=0.5*(Vtomin+Vtomax);
    deltaVto=0.5*(Vtomax-Vtomin);
    if JetEngine ==0
        xTOmode=6*(1+impactVto*((Vto-Vtomean)/deltaVto));
    else
        xTOmode=2*(1+impactVto*((Vto-Vtomean)/deltaVto));
    end
end

%% Landing severity factor
if LAmode == 0
```

```
    Vlamin=50;
    Vlamax=200;
    impactVla=0.05;
    Vlamean=0.5*(Vlamin+Vlamax);
    deltaVla=0.5*(Vlamax-Vlamin);
    xLAmode=8*(1+impactVla*((Vla-Vlamean)/deltaVla));
elseif LAmode == 1
    Vlamin=0;
    Vlamax=200;
    impactVla=0.05;
    Vlamean=0.5*(Vlamin+Vlamax);
    deltaVla=0.5*(Vlamax-Vlamin);
    xLAmode=5*(1+impactVla*((Vla-Vlamean)/deltaVla));
else
    xLAmode=3;
end

%% Occurence: failure rate for each function
cRocketPropu=75.95;
cTOmode=7.38;
cLAmode=10.44;
cJetEngines=1.27;
cPilots=4.96;

%% Output
%Definition of the vectors used in the scalar product
coefs=[cRocketPropu cJetEngines cPilots cTOmode cLAmode];
values=[xRocketPropu xJetEngines xPilots xTOmode xLAmode];

%Output
output=coefs*values';
```

### D.2.7 General Functions

```
function output=designFramework(hMax, nMax, seatPitch, nPAX, nLaunch,...
    numbUnits, programLength, wing, TOmode, LAmode, hTransition,...
    nPilots, horizontalTail, verticalTail, Swing, tcWing, sweepWing, ...
    ARwing, TRwing, dfus, db, ln, la, sweepHTail, ARHTail,...
    sweepVTail, ARVTail, pc, epsilon, Propellant, Tr, ...
    JetEngine, nJet, Tj, BPR, afterburner, TIT)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Deisgn framework: main function that drives the evaluation of a concept
% in terms of performance, weight, life-cycle costs, and safety
% Author: Christopher Frank
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

```

```
%% Design variables
%Constants
LC=90; %Learning curve improvement (%) //Both
year=2015; %Starting year of the program
MmaxJet=2; %Maximum Mach number (-)
tmission=3600; %Mission time (s)
qMax=50000; %Maximum dynamic pressure
hGround=0; %Altitude of the TO
hReentry=30000;
g=9.81;

%% Parameters definition
if wing == 1
    Sref=Swing; %Reference surface area (m^2)
else
    Sref=0.25*pi*db^2; %Fuselage section (m^2)
end
engineDiameter=dfus*0.9;

%% Consistency loop
%Initial guess
nSteps=100;
fuelWeightGuess=0; %Fuel weight (kg)
combTimeGuess=50; %Combustion time (s)
propellantWeightGuess=5e3; %Propellant weight (kg)
fuelWeight=200;
combTime=70;
propellantWeight=6.75e3;
ite=0;

while ((abs(propellantWeight-propellantWeightGuess)/...
        propellantWeightGuess > 0.02)...
        && (abs(combTime-combTimeGuess)/combTimeGuess > 0.02))...
        && (ite<20)
    ite=ite+1;
    propellantWeight=propellantWeightGuess;
    combTime=combTimeGuess;

    %Weight/size
    weight= weightModule(Swing, tcWing, TRwing, ARwing, dfus,...
        wing, nMax, nPilots, nPAX, qMax, tmission,...
        propellantWeight, JetEngine, Tj, nJet, BPR, MmaxJet, afterburner,...
        TIT, fuelWeightGuess, Tr, pc, epsilon, combTime, engineDiameter, ...
        Propellant, seatPitch,  la, ln, db, horizontalTail, verticalTail);
    perfoRocket=weight.rocketPerfo;
    dnac=1.1*weight.jetEngineDiameter; %Nacelle diameter (m)
    lnac=1.1*weight.jetEngineLength; %Nacelle length(m)
    EmptyWeight=weight.EmptyWeight;
    L=weight.length; %length of the vehicle (m)
```

```
    lcyl=L-la; %Length of the cylindrical part of the fuselage (m)
    takeOffWeight=EmptyWeight+fuelWeight+propellantWeight;

    %Parasite drag coefficient
    aeroCd0=aeroModule(Sref, wing, ARwing, sweepWing, tcWing, TRwing,...
        dfus, db, lcyl, la, ln, sweepHTail, ARHTail,...
        dnac, lnac, JetEngine, sweepVTail,...
        ARVTail, horizontalTail, verticalTail);

    WendClimbJet=takeOffWeight;
    if JetEngine == 1
        iteJ=0;
        while ((abs(fuelWeight-fuelWeightGuess)/fuelWeightGuess > 0.01)...
                && (iteJ<15))
            iteJ=iteJ+1;
            fuelWeight=fuelWeightGuess;
            takeOffWeight=EmptyWeight+fuelWeight+propellantWeight;
            jetClimbMod = trajectoryJet(Tj, takeOffWeight, aeroCd0,...
                Swing, nJet, afterburner, TIT, BPR, hGround,...
                hTransition, sweepWing, ARwing, TRwing, tcWing, LAmode,...
                EmptyWeight);
            fuelWeightGuess=jetClimbMod.Dw;
        end
        WendClimbJet=jetClimbMod.WendClimbJet;
    end


    %Rocket trajectory
    trajectory = trajectoryModule(WendClimbJet, aeroCd0, perfoRocket,...
        pc, epsilon, Tr, Sref, nSteps, hMax, hTransition, nMax);

    %Update variables
    propellantWeightGuess=propellantWeight+1.5*(trajectory.propNeeded-...
        propellantWeight);
    combTimeGuess=trajectory.totalTime;

    if ((propellantWeightGuess > 5e5) || (propellantWeightGuess <0))
        errorStruct = struct();
        errorStruct.message = 'Infinite cost';
        errorStruct.identifier = 'Design:infiniteCost';
        error(errorStruct);
    end
end
if ite == 20
    errorStruct = struct();
    errorStruct.message = 'Maximum number of iterations reached';
    errorStruct.identifier = 'Design:maxIter';
    error(errorStruct);
end
```

```
111
112
113 %% Cost %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
114 SOtank=weight.SOtank; %Surface of the oxidizer tank (m^2)
115 SPtank=weight.SPtank; %Surface of the pressurization tank (m^2)
116 O_F=weight.rocketPerfo.O_F; %Oxidizer to fuel ratio (-)
117 Ispv=weight.rocketPerfo.Isp; %Isp (sec)
118 VehicleDryMass=weight.EmptyWeight; %Total weight without fuel (kg)
119 TotalLaunchMass=VehicleDryMass+weight.fuelPropellantWeight; %TOGW (kg)
120 rocketEngineEmptyWeight=weight.rocketEngineWeight;%Rocket engine weight(kg)
121 WdryjetEng=weight.jetEngineWeight; %Jet engine weight (kg)
122
123 outputCost=costModule(wing, nLaunch, programLength, numbUnits, tmission,...
124     nPilots, nPAX, TOmode, year, LC, nJet, JetEngine, Ispv,...
125     propellantWeight, Propellant, O_F, Tr, SOtank, SPtank,...
126     TotalLaunchMass, VehicleDryMass, rocketEngineEmptyWeight,...
127     afterburner, WdryjetEng, TIT, Tj, fuelWeight);
128 cost = outputCost.TotalCost;
129
130 %% Take-off performance %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
131 if TOmode == 1
132     if JetEngine == 1
133         T07=Tj;
134     else
135         T07=Tr;
136     end
137     TOperfo=TOModule(sweepWing, Swing, TotalLaunchMass, T07);
138     Vto=TOperfo.TOSpeed;
139     TOFL=TOperfo.TOFL;
140 else
141     Vto=0;
142     TOFL=0;
143 end
144
145 %% Landing performance %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
146 if LAmode < 1.5
147     LAperfo=LandingModule(sweepWing, Swing, VehicleDryMass);
148     Vla=LAperfo.landingSpeed;
149     LAFL=LAperfo.TOFL;
150 else
151     Vla=0;
152     LAFL=0;
153 end
154
155 runwayLength=max(LAFL,TOFL);
156
157 %%Weightlessness time
158 tWLN = sqrt(2/g)*(sqrt(hMax-trajectory.hend)+sqrt(hMax-hReentry));
159
```

```
%% Safety
safety=safetyModule(pc, Propellant, LAmode, TOmode, nJet, nPilots,...
    JetEngine, Tr, Vla, Vto, afterburner);

output.runwayLength=runwayLength;
output.cost=cost;
output.safety=safety;
output.extra=outputCost;
output.extra.Mtot=takeOffWeight;
output.extra.hend = trajectory.hend;
output.extra.Mend = trajectory.Mend;
output.extra.tWLN = tWLN;

end
```

```
function output = funcAtm(h)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% General: Calculate atmospheric parameters
% Authors: Christopher Frank and Clemence Tyl
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

p0=101325;

%Temperature
if (h <= 11000)
    T = 15-0.0065*h;
elseif(11000 < h)&&(h<=20100)
    T = -56.5;
elseif(20100 < h)&&(h<=32200)
    T = -56.5+0.001*(h-20100);
elseif(32200 < h)&&(h<=47300)
    T = -44.5+0.0028*(h-32200);
elseif(47300 < h)&&(h<=52400)
    T = -2.5;
elseif(52400 < h)&&(h<=61600)
    T = -2.5-0.002*(h-52400);
elseif(61600 < h)&&(h<=80000)
    T = -20.5-0.003913043478*(h-61600);
elseif(80000 < h)&&(h<=93000)
    T = -92.5;
else
    T = (92.5*h-92.5*125000)/(125000-93000);
end

%Pressure
if (h <= 10000)
```

```
    p = p0*(1-0.0065/288.15*h)^5.255;
else
    p = 126000*exp(-0.0001560*h);
end

%Temperature conversion to SI (K)
output.T = T+273.15;

%Pressure
output.p = p;

%Density
rho = p/(287.058*(T+273.15));
output.rho = rho ;

%Acceleration of gravity
g = 9.81/(1+2*h/6378000);
output.g = g;

%Dynamic viscosity
%(http://www.arc.vt.edu/ansys_help/flu_ug/x1-7350009.4.2.html)
mu0=1.716*1e-5;
T0=273.11;
S=110.56;
mu=mu0*((T+273.15)/T0)^(3/2)*(T0+S)/(T+S);
output.mu=mu;

end
```

```
function output = MtoV(M, h)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% General: Conversion from Mach number to speed
% Author: Christopher Frank
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

atm=funcAtm(h);
T=atm.T;
gamma=1.4;
R=287.058;

output=M*sqrt(gamma*R*T);

end
```

```
function output = VtoM(V, h)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% General: Conversion from speed to Mach number
% Author: Christopher Frank
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

atm=funcAtm(h);
T=atm.T;
gamma=1.4;
R=287.058;

output=V/sqrt(gamma*R*T);

end
```

### *D.3 Optimization Module*

The optimization module is based on the NSGA-II code developed by Lin Song from the Aerospace Structural Dynamics Research Laboratory of the College of Astronautics, Northwestern Polytechnical University, China [407, 408]. It has been modified and completed to match the capabilities described in Section 7.2.

```
function opt = callOutputfuns(opt, state, pop, type)
% Function: opt = callOutputfuns(opt, state, pop, type)
% Description: Call output function(if exist).
% Parameters:
%   type : output type.
%       -1 = the last call (close file for instance)
%       other values(or no exist) = normal output
%
%         LSSSSWC, NWPU
%    Revision: 1.1  Data: 2011-07-13
%*********************************************************************

if(nargin <= 3)
    type = 0;   % normal output
end

if( ~isempty(opt.outputfuns) )
    fun = opt.outputfuns{1};
    opt = fun(opt, state, pop, type, opt.outputfuns{2:end});

end
```

```
function pop = crossoverOp(opt, pop, state)
% Function: pop = crossoverOp(opt, pop, state)
% Description: Crossover operator. All of the individuals would be do
% crossover, but only "crossoverFraction" of design variables of an
% individual would changed.
%
%         LSSSSWC, NWPU
%    Revision: 1.1  Data: 2011-07-13
%*********************************************************************

%*********************************************************************
% 1. Check for the parameters
%*********************************************************************
% determine the crossover method
strfun = lower(opt.crossover{1});
numOptions = length(opt.crossover) - 1;
[crossoverOpt{1:numOptions}] = opt.crossover{2:end};

switch( strfun )
    case 'intermediate'
        fun = @crsIntermediate;
    otherwise
        error('NSGA2:CrossoverOpError', 'No support crossover operator!');
end

nVar = opt.numVar;

% "auto" crossover fraction
if( ischar(opt.crossoverFraction) )
    if( strcmpi(opt.crossoverFraction, 'auto') )
        fraction = 2.0 / nVar;
    else
        error('NSGA2:CrossoverOpError', strcat('The "crossoverFraction"',...
            'parameter should be scalar or "auto" string.'));
    end
else
    fraction = opt.crossoverFraction;
end


for ind = 1:2:length(pop)    % Popsize should be even number
        % Create children
        [child1, child2] = fun( pop(ind), pop(ind+1),...
            fraction, crossoverOpt );

        % Round
        for v = 1:nVar
            if( opt.vartype(v) == 2)
                child1.var(v) = round( child1.var(v) );
```

```
                child2.var(v) = round( child2.var(v) );
            end
        end

        % Bounding limit
        child1.var = varlimit(child1.var, opt.lb, opt.ub);
        child2.var = varlimit(child2.var, opt.lb, opt.ub);

        pop(ind)     = child1;
        pop(ind+1)   = child2;

end



function [child1, child2] = crsIntermediate(parent1, parent2, ...
    fraction, options)
% Function: [child1, child2] = crsIntermediate(parent1, parent2,
% fraction, options)
% Description: (For real coding) Intermediate crossover.
% (Same as Matlab's crossover operator)
%       child = parent1 + rand * Ratio * ( parent2 - parent1)
% Parameters:
%   fraction : crossover fraction of variables of an individual
%   options = ratio
%
%         LSSSSWC, NWPU
%    Revision: 1.1  Data: 2011-07-13
%*********************************************************************


if( length(options)~=1 || ~isnumeric(options{1}))
    error('NSGA2:CrossoverOpError', 'Crossover operator parameter error!');
end

ratio = options{1};

child1 = parent1;
child2 = parent2;

nVar = length(parent1.var);
crsFlag = rand(1, nVar) < fraction;

randNum = rand(1,nVar);     % uniformly distribution

child1.var = parent1.var + crsFlag .* randNum .* ratio .* (parent2.var...
    - parent1.var);
child2.var = parent2.var - crsFlag .* randNum .* ratio .* (parent2.var...
    - parent1.var);
```

```
function [pop, state] = evaluate(opt, pop, state, varargin)
% Function: [pop, state] = evaluate(opt, pop, state, varargin)
% Description: Evaluate the objective functions of each individual in the
%   population.
%
%         LSSSSWC, NWPU
%    Revision: 1.0  Data: 2011-04-20
%*************************************************************************

N = length(pop);
allTime = zeros(N, 1); %allTime: use to calculate average evaluation times

%*************************************************************************
% Evaluate objective function in parallel
%*************************************************************************
if( strcmpi(opt.useParallel, 'yes') == 1 )

    parfor i = 1:N
        fprintf(strcat('\nEvaluating the objective function...',...
            'Generation: %d / %d , Individual: %d / %d \n'),...
            state.currentGen, opt.maxGen, i, N);
        [pop(i), allTime(i)] = evalIndividual(pop(i), opt.objfun,...
            varargin{:});
    end

    %*********************************************************************
    % Evaluate objective function in serial
    %*********************************************************************
else
    for i = 1:N
        fprintf(strcat('\nEvaluating the objective function...',...
            ' Generation: %d / %d , Individual: %d / %d \n'),...
            state.currentGen, opt.maxGen, i, N);
        [pop(i), allTime(i)] = evalIndividual(pop(i),...
            opt.objfun, varargin{:});
    end
end

%*************************************************************************
% Statistics
%*************************************************************************
state.avgEvalTime   = sum(allTime) / length(allTime);
state.evaluateCount = state.evaluateCount + length(pop);



function [indi, evalTime] = evalIndividual(indi, objfun, varargin)
% Function: [indi, evalTime] = evalIndividual(indi, objfun, varargin)
```

```
% Description: Evaluate one objective function.
%
%         LSSSSWC, NWPU
%    Revision: 1.1  Data: 2011-07-25
%*************************************************************************
tStart = tic;
%objfun( indi.var, varargin{:} )
[y, cons, extra] = objfun( indi.var, varargin{:} );
evalTime = toc(tStart);

% Save the objective values and constraint violations
indi.obj = y;
indi.extra = extra;
if( ~isempty(indi.cons) )
    idx = find( cons );
    if( ~isempty(idx) )
        indi.nViol = length(idx);
        indi.violSum = sum( abs(cons) );
    else
        indi.nViol = 0;
        indi.violSum = 0;
    end
end
```

```
function [gen,count] = EvolutionaryAlgorithm(maxArchi, old_gen, count, ...
    popsize,numObj, numExtra)
%*************************************************************************
% Output the number of generations for the next iteration for each
% architecture given the number of points on the pareto frontier per
% architecture
%*************************************************************************

T = 5;
nbObj = ones(1,maxArchi)*numObj;
nbDV = zeros(1,maxArchi);
nbCstn = zeros(1,maxArchi);
nbExtra = ones(1,maxArchi)*numExtra;
for j = 1:maxArchi
    [nbDV(j), nbCstn(j)] = Architecture(j);
end

%% SCAN FILES
for j = 1:maxArchi
    fid=fopen(sprintf('optresult%d.txt',j),'r');
    for i = 1:20
        fgetl(fid);
```

```
    end
    A=fscanf(fid,'%f',[nbDV(j)+nbObj(j)+nbCstn(j)+nbExtra(j) inf]).';
    V{j} = A;
    fclose(fid);
end

%% PARETO POINTS
optpt=[];
for j=1:maxArchi
    optpt = vertcat(optpt,V{j}(:,nbDV(j)+1:nbDV(j)+nbObj(j)));
    [~, lign] = prtp(optpt);
end

%% NUMBER OF POINTS PER ARCHITECTURE
a = zeros(1,maxArchi);
for z = 1:length(lign)
    for j = 1:maxArchi
        if lign(z) <= popsize*j && lign(z) > popsize*(j-1)
            a(j) = a(j)+1;
        end
    end
end

for j = 1:maxArchi
    fprintf('Number of Pareto points in Architecture %d:',j)
    disp(a(j))
end

%% NUMBER OF GENERATION FOR NEXT ITERATION
gen = zeros(1,maxArchi);
for z = 1:maxArchi
    gen(z) = a(z)/max(a)*max(old_gen);
    if gen(z) == 0;
        p = 1-exp(-count(z)/T);
        count(z) = count(z)+1;
        if rand() <= p;
            gen(z) = max(old_gen);
            count(z) = 0;
        end
    end
end

gen = round(gen);
end
```

```
function nextpop = extractPop(opt, combinepop)
```

```
% Function: nextpop = extractPop(opt, combinepop)
% Description: Extract the best n individuals in 'combinepop'(population
%   size is 2n).
%
%         LSSSSWC, NWPU
%    Revision: 1.1  Data: 2011-07-12
%*************************************************************************

popsize = length(combinepop) / 2;
nextpop = combinepop(1:popsize);    %just for initializing

rankVector = vertcat(combinepop.rank);

n = 0;          % individuals number of next population
rank = 1;       % current rank number
idx = find(rankVector == rank);
numInd = length(idx);       % number of individuals in current front
while( n + numInd <= popsize )
    nextpop( n+1 : n+numInd ) = combinepop( idx );

    n = n + numInd;
    rank = rank + 1;

    idx = find(rankVector == rank);
    numInd = length(idx);
end

% If the number of individuals in the next front plus the number of
% individuals in the current front is greater than the population size,
% then select the best individuals by corwding distance (NSGA-II)
% or preference distance(R-NSGA-II).
if( n < popsize )
    if(~isempty(opt.refPoints))
        prefDistance = vertcat(combinepop(idx).prefDistance);
        prefDistance = [prefDistance, idx];
        prefDistance = sortrows( prefDistance, 1);
        % Select the individuals with smallest preference distance
        idxSelect  = prefDistance( 1:popsize-n, 2);
        nextpop(n+1 : popsize) = combinepop(idxSelect);
    else
        distance = vertcat(combinepop(idx).distance);
        distance = [distance, idx];
        % Sort the individuals in descending order of crowding distance
        % in the front.
        distance = flipud( sortrows( distance, 1) );
        % Select the (popsize-n) individuals with largest crowding distance
        idxSelect  = distance( 1:popsize-n, 2);
        nextpop(n+1 : popsize) = combinepop(idxSelect);
    end
```

```
end
```

```
function pop = initpop(opt, pop, varargin)
% Function: pop = initpop(opt, pop, varargin)
% Description: Initialize population.
% Syntax:
%   pop = initpop(opt, pop)
%     (default) Create a random initial population with a uniform
%      distribution.
%
%   pop = initpop(opt, pop, 'pop.txt')
%     Load population from exist file and use the last population. If
%     the popsize less than the current popsize, then random numbers will
%     used to fill the population.
%
%   pop = initpop(opt, pop, 'pop.txt', ngen)
%     Load population from file with specified generation.
%
%   pop = initpop(opt, pop, oldresult)
%     Specify exist result structure.
%
%   pop = initpop(opt, pop, oldresult, ngen)
%     Specify exist result structure and the population which will be used.
%
% Parameters:
%   pop : an empty population
%
%         LSSSSWC, NWPU
%    Revision: 1.1  Data: 2011-07-01
%*************************************************************************


%*************************************************************************
% 1. Identify parameters
%*************************************************************************
method = 'uniform';
if(nargin >= 3)
    if( ischar(varargin{1}) )
        method = 'file';
    elseif( isstruct(varargin{1}) )
        method = 'existpop';
    end
end

%*************************************************************************
% 2. Initialize population with different methods
```

```
%%************************************************************************
if( strcmpi(method, 'uniform'))
    pop = initpopUniform(opt, pop);
elseif(strcmpi(method, 'file'))
%    fprintf('...Initialize population from file "%s"\n', varargin{1});
    pop = initpopFromFile(opt, pop, varargin{:});
elseif(strcmpi(method, 'existpop'))
%    fprintf('...Initialize population from specified result.\n');
    pop = initpopFromExistResult(opt, pop, varargin{:});
end



function pop = initpopFromFile(opt, pop, varargin)
% Function: pop = initpopFromFile(opt, pop, varargin)
% Description: Load population from specified population file.
% Syntax:
%   pop = initpop(opt, pop, 'pop.txt')
%   pop = initpop(opt, pop, 'pop.txt', ngen)
%
%    Copyright 2011 by LSSSSWC
%    Revision: 1.0  Data: 2011-07-01
%%************************************************************************
fileName = varargin{1};

oldResult = loadpopfile(fileName);
fprintf('initpopFromFile')
pop = initpopFromExistResult(opt, pop, oldResult, varargin{2:end});



function pop = initpopFromExistResult(opt, pop, varargin)
% Function: pop = initpopFromExistResult(opt, pop, varargin)
% Description: Load population from exist result structure.
% Syntax:
%   pop = initpop(opt, pop, oldresult)
%   pop = initpop(opt, pop, oldresult, ngen)
%
%    Copyright 2011 by LSSSSWC
%    Revision: 1.0  Data: 2011-07-01
%%************************************************************************

% 1. Verify param
fprintf('initpopFromExistResult')
oldresult = varargin{1};
if( ~isstruct(oldresult) || ~isfield(oldresult, 'pops') )
```

```
94      error('NSGA2:InitPopError',...
95          'The result structure specified is not correct!');
96  end
97
98
99  oldpops = oldresult.pops;
100 ind = oldpops(1,1);     % individual used to verify optimization param
101 if( opt.numVar ~= length(ind.var) || ...
102     opt.numObj ~= length(ind.obj) || ...
103     opt.numCons ~= length(ind.cons) )
104     error('NSGA2:InitPopError', ...
105         strcat('The specified optimization result is',...
106         'not for current optimization model!'));
107 end
108 clear ind
109
110
111 % 2. Determine which population would be used
112 ngen = 0;
113 if( nargin >= 4)
114     ngen = varargin{2};
115 end
116
117 maxGen = size(oldpops, 1);
118 if(ngen == 0)
119     ngen = maxGen;
120 elseif(ngen > maxGen)
121     warning('NSGA2:InitPopWarning', ...
122         'The specified generation "%d" does not exist, use "%d" instead.',...
123         ngen, maxGen);
124     ngen = maxGen;
125 end
126
127
128 % 3. Create initial population
129 popsizeOld = size(oldpops, 2);
130 popsizeNew = opt.popsize;
131
132 if( popsizeNew <= popsizeOld )      % a) All from old pop
133     for i = 1:popsizeNew
134         pop(i).var = oldpops(ngen, i).var;
135     end
136 else          % b) Use random individuals to fill the population
137     for i = 1:popsizeOld
138         pop(i).var = oldpops(ngen, i).var;
139     end
140     pop(popsizeOld+1:end) = initpopUniform(opt, pop(popsizeOld+1:end));
141 end
142
```

```
143
144
145
146 function pop = initpopUniform(opt, pop)
147 % Function: pop = initpopUniform(opt, pop)
148 % Description: Initialize population using random number
149 %
150 %    Copyright 2011 by LSSSSWC
151 %    Revision: 1.0  Data: 2011-07-01
152 %*********************************************************************
153
154 nVar = opt.numVar;
155 type = opt.vartype;
156
157 lb = opt.lb;
158 ub = opt.ub;
159
160 popsize = length(pop);
161 for i = 1:popsize
162     var = lb + rand(1, nVar) .* (ub-lb);
163
164     % if desing variable is integer, round to the nearest integer
165     for v = 1:nVar
166         if( type(v) == 2)
167             var(v) = round(var(v));
168         end
169     end
170
171     % limit in the lower and upper bound
172     var = varlimit(var, lb, ub);
173
174     pop(i).var = var;
175
176 end
```

```
1  function result = loadpopfile(fileName)
2  % Function: result = loadpopfile(fileName)
3  % Description: Load population file which generated by last optimization.
4  % Syntax:
5  %       oldresult = loadpopfile('populations.txt');
6  % Return:
7  %
8  %    Copyright 2011 by LSSSSWC
9  %    Revision: 1.0  Data: 2011-07-01
10 %*********************************************************************
11
```

```
12
13 %*************************************************************************
14 % Open the populations file
15 %*************************************************************************
16 if( ~ischar(fileName))
17     error('NSGA2:LoadPopError',...
18         'The fileName parameter should be string!');
19 end
20
21 fid = fopen(fileName, 'r');
22 if(fid==-1)
23     error('NSGA2:LoadPopError',...
24         'The populations file "%s" could not opened!', fileName);
25 end
26 %fprintf('...Loading the population file "%s"......\n', fileName);
27
28
29
30 %*************************************************************************
31 % Read file head
32 %*************************************************************************
33 popsize = 1;
34 maxGen  = 1;
35 nVar  = 0;
36 nObj  = 0;
37 nCons = 0;
38 nExtra = 0;
39 fieldNames = {''};
40
41 strLine = fgetl(fid);
42 if( ~ischar(strLine) || strcmp(strLine, '#NSGA2')==0 )
43     error('NSGA2:PopFileError', ...
44         strcat('The population file "%s" is not a Nsga2',...
45         'populations file! Line \n%s\n'), ...
46         fileName, strLine);
47 end
48
49 strLine = fgetl(fid);
50 while( ischar(strLine) && strcmp(strLine, '#end')==0)
51     token = textscan(strLine, '%s');
52     keyword = strtrim(token{1}{1});
53     switch keyword
54         case 'popsize'
55             popsize = str2double(token{1}{2});
56         case 'maxGen'
57             maxGen = str2double(token{1}{2});
58         case 'numVar'
59             nVar = str2double(token{1}{2});
60         case 'numObj'
```

```
61              nObj = str2double(token{1}{2});
62          case 'numCons'
63              nCons = str2double(token{1}{2});
64          case 'numExtra'
65              nExtra = str2double(token{1}{2});
66          case 'stateFieldNames'
67              nfield = length(token{1});
68              fieldNames = cell(nfield - 1, 1);
69              [fieldNames{:}] = token{1}{2:end};
70          otherwise
71              warning('NSGA2:PopFileError',...
72                  'No support state keyword: "%s"', token{1}{1});
73      end
74
75      strLine = fgetl(fid);
76  end
77
78
79  %************************************************************************
80  % Initialize the result structure
81  %************************************************************************
82  pop = repmat( struct(...
83      'var', zeros(1,nVar), ...
84      'obj', zeros(1,nObj), ...
85      'cons', zeros(1,nCons),...
86      'extra', zeros(1,nExtra)),...
87      [1,popsize]);
88
89  % state: optimization state of one generation
90  state = struct();
91  for i = 1:length(fieldNames)
92      state.(fieldNames{i}) = 0;
93  end
94  % each row is the population of one generation
95  result.pops     = repmat(pop, [maxGen, 1]);
96  % each row is the optimizaiton state of one generation
97  result.states   = repmat(state, [maxGen, 1]);
98  clear i fieldNames pop
99
100
101 %************************************************************************
102 % Parse the populations file
103 %************************************************************************
104 lastGen = 0;    % The last generation which has correct datas.
105 try
106     strLine = fgetl(fid);
107     while ischar(strLine)
108         %****************************************************************
109         % 1. Skip empty lines
```

```
110         strLine = strtrim(strLine);
111         if( isempty(strLine)  )
112             strLine = fgetl(fid);   % read new line
113             continue;
114         end
115 
116         %*********************************************************************
117         % 2. The first line of one generation
118         if( strcmp(strLine(1:11), '#Generation') == 1)
119             % Only the 'ngen' is needed
120             ngen = sscanf(strLine(12:end), ' %d');
121         else
122             error('NSGA2:PopFileError',...
123                 'The population file format Error Line \n%s\n', strLine);
124         end
125 
126 
127         %*********************************************************************
128         % 3. Read optimization states
129         strLine = fgetl(fid);
130         while( ischar(strLine) && strcmp(strLine, '#end')==0 )
131             token = textscan(strLine, '%s%f');
132             result.states(ngen).(token{1}{1}) = token{1,2};
133 
134             strLine = fgetl(fid);
135         end
136 
137 
138         %*********************************************************************
139         % 4. Read population
140         strLine = fgetl(fid);
141         val = fscanf(fid, '%f');
142         ncols = nVar+nObj+nCons+nExtra;
143         nrows = popsize;
144         if( length(val) ~= ncols*nrows )
145             error('NSGA2:PopFileError',...
146                 'File error when read population datas! ');
147         end
148 
149         val = reshape(val, ncols, nrows)'; %reshape the vector column-wise
150         for i = 1:popsize
151             result.pops(ngen, i).var = val(i, 1:nVar);
152             result.pops(ngen, i).obj = val(i, (nVar+1):(nVar+nObj));
153             result.pops(ngen, i).cons= val(i, ...
154                 (nVar+nObj+1):nVar+nObj+nCons);
155             result.pops(ngen, i).extra= val(i, (nVar+nObj+nCons+1):end);
156         end
157         %*********************************************************************
158         % Read next line
```

```
159         lastGen = ngen;
160         strLine = fgetl(fid);
161 
162     end
163 catch exception
164     id = exception.identifier;
165     msg = [exception.message, 'File = ', fileName];
166     warning(id, msg);
167     % If the file is wrong at the end, return the correct data.
168 end
169 
170 
171 %%************************************************************************
172 % Do some clean works
173 %%************************************************************************
174 
175 % Delete unused datas
176 result.pops(lastGen+1:end, :) = [];
177 result.states(lastGen+1:end)  = [];
178 
179 % Close file
180 fclose(fid);
```

```
1  function pop = mutationOp(opt, pop, state)
2  % Function: pop = mutationOp(opt, pop, state)
3  % Description: Mutation Operator. All of the individuals would do mutation,
4  % but only "mutationFraction" of design variables of an individual would
5  % changed.
6  %
7  %         LSSSSWC, NWPU
8  %    Revision: 1.1  Data: 2011-07-13
9  %************************************************************************
10 
11 %************************************************************************
12 % 1. Check for the parameters
13 %************************************************************************
14 % mutation method
15 strfun = lower(opt.mutation{1});
16 numOptions = length(opt.mutation) - 1;
17 [mutationopt{1:numOptions}] = opt.mutation{2:end};
18 
19 switch (strfun)
20     case 'gaussian'
21         fun = @mutationGaussian;
22     otherwise
23         error('NSGA2:MutationOpError', 'No support mutation operator!');
```

```
24 end
25
26 nVar = opt.numVar;
27
28 % "auto" mutation fraction
29 if( ischar(opt.mutationFraction) )
30     if( strcmpi(opt.mutationFraction, 'auto') )
31         fraction = 2.0 / nVar;
32     else
33         error('NSGA2:MutationOpError',...
34             strcat('The "mutationsFraction" parameter should',...
35         'be scalar or "auto" string.'));
36     end
37 else
38     fraction = opt.mutationFraction;
39 end
40
41
42 % All of the individual would be modified, but only 'mutationFraction' of
43 % design variables for an individual would be changed.
44 for ind = 1:length(pop)
45         child = fun( pop(ind), opt, state, fraction, mutationopt);
46
47         % Rounding for integer variables
48         for v = 1:nVar
49             if( opt.vartype(v) == 2)
50                 child.var(v) = round( child.var(v) );
51             end
52         end
53
54         child.var = varlimit(child.var, opt.lb, opt.ub);
55
56         pop(ind) = child;
57 end
58
59
60
61 function child = mutationGaussian( parent, opt, state, fraction, options)
62 % Function: child = mutationGaussian( parent, opt, state, fraction,
63 % options)
64 % Description: Gaussian mutation operator. Reference Matlab's help :
65 %   Genetic Algorithm Options :: Options Reference (Global Optimization
66 %   Toolbox)
67 % Parameters:
68 %   fraction : mutation fraction of variables of an individual
69 %   options{1} : scale. This paramter should be large enough for interger
70 %   variables to change from one to another.
71 %   options{2} : shrink
72 % Return:
```

```
%
%         LSSSSWC, NWPU
%    Revision: 1.1  Data: 2011-07-13
%*************************************************************************


%*************************************************************************
% 1. Verify the parameters.
%*************************************************************************
if( length(options)~=2)
    error('NSGA2:MutationOpError', 'Mutation operator parameter error!');
end


%*************************************************************************
% 2. Calc the "scale" and "shrink" parameter.
%*************************************************************************
scale = options{1};
shrink = options{2};
scale = scale - shrink * scale * state.currentGen / opt.maxGen;

lb = opt.lb;
ub = opt.ub;
scale = scale * (ub - lb);


%*************************************************************************
% 3. Do the mutation.
%*************************************************************************
child = parent;
numVar = length(child.var);
for i = 1:numVar
    if(rand() < fraction)
        child.var(i) = parent.var(i) + scale(i) * randn();
    end
end
```

```
function [opt, pop] = ndsort(opt, pop)
% Function: [opt, pop] = ndsort(pop)
% Description: Fast non-dominated sort.
%
%         LSSSSWC, NWPU
%    Revision: 1.4  Data: 2011-07-26
%*************************************************************************

%*************************************************************************
```

```
% 1. Initialize variables
%   indi.np number of individuals which dominate this individual
%   indi.sp(:): a set of individuals that this individual dominate
%*************************************************************************
N = length(pop);    %popsize
ind = repmat(struct('np',0, 'sp', []),[1,N]);

for i = 1:N
    pop(i).rank = 0;
    pop(i).distance = 0;
    pop(i).prefDistance = 0;
end


%*************************************************************************
% 2. fast non-dominated sort
%*************************************************************************
% Calculate the domination matrix for improving the efficiency.

nViol   = zeros(N, 1);
violSum = zeros(N, 1);
for i = 1:N
    nViol(i)    = pop(i).nViol;
    violSum(i)  = pop(i).violSum;
end
% nViol   = vertcat(pop(:).nViol);
% violSum = vertcat(pop(:).violSum);
obj     = vertcat(pop(:).obj);
% domination matrix for efficiency
domMat  = calcDominationMatrix(nViol, violSum, obj);

% Compute np and sp of each indivudal
for p = 1:N-1
    for q = p+1:N
        if(domMat(p, q) == 1)          % p dominate q
            ind(q).np = ind(q).np + 1;
            ind(p).sp = [ind(p).sp , q];
        elseif(domMat(p, q) == -1)     % q dominate p
            ind(p).np = ind(p).np + 1;
            ind(q).sp = [ind(q).sp , p];
        end
    end
end


% The first front(rank = 1)
front(1).f = [];    % There are only one field 'f' in structure 'front'.
                    % This is intentional because the number of individuals
                    % in the front is difference.
```

```
for i = 1:N
    if( ind(i).np == 0 )
        pop(i).rank = 1;
        front(1).f = [front(1).f, i];
    end
end

% Calculate pareto rank of each individuals, viz., pop(:).rank
fid = 1;        %pareto front ID
while( ~isempty(front(fid).f) )
    Q = [];
    for p = front(fid).f
        for q = ind(p).sp
            ind(q).np = ind(q).np -1;
            if( ind(q).np == 0 )
                pop(q).rank = fid+1;
                Q = [Q, q];
            end
        end
    end
    fid = fid + 1;

    front(fid).f = Q;
end
front(fid) = [];    % delete the last empty front set



%**************************************************************************
% 3. Calculate the distance
%**************************************************************************
if(isempty(opt.refPoints))
    pop = calcCrowdingDistance(opt, pop, front);
else
    [opt, pop] = calcPreferenceDistance(opt, pop, front);
end




function domMat = calcDominationMatrix(nViol, violSum, obj)
% Function: domMat = calcDominationMatrix(nViol, violSum, obj)
% Description: Calculate the domination maxtir which specified the
% domination releation between two individual using constrained-domination.
%
% Return:
%   domMat(N,N) : domination matrix
%       domMat(p,q)=1   : p dominates q
```

```
%       domMat(p,q)=-1 : q dominates p
%       domMat(p,q)=0  : non dominate
%
%    Copyright 2011 by LSSSSWC
%    Revision: 1.0  Data: 2011-07-13
%*************************************************************************

N       = size(obj, 1);
numObj  = size(obj, 2);

domMat  = zeros(N, N);

for p = 1:N-1
    for q = p+1:N
        %*********************************************************************
        % 1. p and q are both feasible
        %*********************************************************************
        if(nViol(p) == 0 && nViol(q)==0)
            pdomq = false;
            qdomp = false;
            for i = 1:numObj
                % objective function is minimization!
                if( obj(p, i) < obj(q, i) )
                    pdomq = true;
                elseif(obj(p, i) > obj(q, i))
                    qdomp = true;
                end
            end

            if( pdomq && ~qdomp )
                domMat(p, q) = 1;
            elseif(~pdomq && qdomp )
                domMat(p, q) = -1;
            end
        %*********************************************************************
        % 2. p is feasible, and q is infeasible
        %*********************************************************************
        elseif(nViol(p) == 0 && nViol(q)~=0)
            domMat(p, q) = 1;
        %*********************************************************************
        % 3. q is feasible, and p is infeasible
        %*********************************************************************
        elseif(nViol(p) ~= 0 && nViol(q)==0)
            domMat(p, q) = -1;
        %*********************************************************************
        % 4. p and q are both infeasible
        %*********************************************************************
        else
            if(violSum(p) < violSum(q))
```

```
157                 domMat(p, q) = 1;
158             elseif(violSum(p) > violSum(q))
159                 domMat(p, q) = -1;
160             end
161         end
162     end
163 end
164
165 domMat = domMat - domMat';
166
167
168
169
170
171 function [opt, pop] = calcPreferenceDistance(opt, pop, front)
172 % Function: [opt, pop] = calcPreferenceDistance(opt, pop, front)
173 % Description: Calculate the 'preference distance' used in R-NSGA-II.
174 % Return:
175 %   opt : This structure may be modified only when opt.refUseNormDistance
176 %   =='ever'.
177 %
178 %    Copyright 2011 by LSSSSWC
179 %    Revision: 1.1  Data: 2011-07-26
180 %*************************************************************************
181
182 %*************************************************************************
183 % 1. Initialization
184 %*************************************************************************
185 numObj = length( pop(1).obj );  % number of objectives
186
187 refPoints = opt.refPoints;
188 refWeight = opt.refWeight;      % weight factor of objectives
189 if(isempty(refWeight))
190     refWeight = ones(1, numObj);
191 end
192 epsilon = opt.refEpsilon;
193 numRefPoint = size(refPoints, 1);
194
195 % Determine the normalized factors
196 % bUseFrontMaxMin : If use the maximum and minimum value in the front as
197 % normalized factor.
198 bUseFrontMaxMin = false;
199 if( strcmpi(opt.refUseNormDistance, 'ever') )
200     % 1) Find possiable (not current population) maximum and minimum value
201     %     of each objective.
202     obj = vertcat(pop.obj);
203     if( ~isfield(opt, 'refObjMax_tmp') )
204         opt.refObjMax_tmp = max(obj);
205         opt.refObjMin_tmp = min(obj);
```

```
206     else
207         objMax = max(obj);
208         objMin = min(obj);
209         for i = 1:numObj
210             if(opt.refObjMax_tmp(i) < objMax(i))
211                 opt.refObjMax_tmp(i) = objMax(i);
212             end
213             if(opt.refObjMin_tmp(i) > objMin(i))
214                 opt.refObjMin_tmp(i) = objMin(i);
215             end
216         end
217         clear objMax objMin
218     end
219     objMaxMin = opt.refObjMax_tmp - opt.refObjMin_tmp;
220     clear obj
221 elseif( strcmpi(opt.refUseNormDistance, 'front') )
222     % 2) Do not use normalized Euclidean distance.
223     bUseFrontMaxMin = true;
224 elseif( strcmpi(opt.refUseNormDistance, 'no') )
225     % 3) Do not use normalized Euclidean distance.
226     objMaxMin = ones(1,numObj);
227 else
228     % 3) Error
229     error('NSGA2:ParamError', ...
230         strcatz('No support parameter: options.refUseNormDistance',...
231         '="%s", only "yes" or "no" are supported'),...
232         opt.refUseNormDistance);
233 end
234
235
236 %%************************************************************************
237 % 2. Calculate preference distance pop(:).prefDistance
238 %%************************************************************************
239 for fid = 1:length(front)
240     % Step1: Calculate the weighted Euclidean distance in each front
241     % idxFront : index of individuals in current front
242     idxFront = front(fid).f;
243     % numInd : number of individuals in current front
244     numInd = length(idxFront);
245     % popFront : individuals in front fid
246     popFront = pop(idxFront);
247
248     % objFront : the whole objectives of all individuals
249     objFront = vertcat(popFront.obj);
250
251     if(bUseFrontMaxMin)
252         % objMaxMin : the normalized factor in current front
253         objMaxMin = max(objFront) - min(objFront);
254     end
```

```
255
256     % normDistance : weighted normalized Euclidean distance
257     normDistance = calcWeightNormDistance(objFront, refPoints, ...
258         objMaxMin, refWeight);
259
260     % Step2: Assigned preference distance
261     prefDistanceMat = zeros(numInd, numRefPoint);
262     for ipt = 1:numRefPoint
263         [~,ix] = sort(normDistance(:, ipt));
264         prefDistanceMat(ix, ipt) = 1:numInd;
265     end
266     prefDistance = min(prefDistanceMat, [], 2);
267     clear ix
268
269
270     % Step3: Epsilon clearing strategy
271     % idxRemain : index of individuals which were not processed
272     idxRemain = 1:numInd;
273     while(~isempty(idxRemain))
274         % 1. Select one individual from remains
275         objRemain = objFront( idxRemain, :);
276         selIdx = randi( [1,length(idxRemain)] );
277         selObj = objRemain(selIdx, :);
278
279         % 2. Calc normalized Euclidean distance
280         % distanceToSel: normalized Euclidean distance to the selected
281         % points
282         distanceToSel = calcWeightNormDistance(objRemain, selObj,...
283             objMaxMin, refWeight);
284
285
286         % 3. Process the individuals within a epsilon-neighborhood
287         idx = find( distanceToSel <= epsilon ); % idx: index in idxRemain
288         if(length(idx) == 1)    % the only individual is the selected one
289             idxRemain(selIdx)=[];
290         else
291             for i=1:length(idx)
292                 if( idx(i)~=selIdx )
293                     %idx is the index in idxRemain vector
294                     idInIdxRemain = idx(i);
295                     id = idxRemain(idInIdxRemain);
296
297                     % Increase the preference distance to discourage the
298                     % individuals to remain in the selection.
299                     prefDistance(id) = prefDistance(id) + round(numInd/2);
300                 end
301             end
302             idxRemain(idx) = [];
303         end
```

```
304
305     end
306
307     % Save the preference distance
308     for i=1:numInd
309         id = idxFront(i);
310         pop(id).prefDistance = prefDistance(i);
311     end
312 end
313
314
315 function distance = calcWeightNormDistance(points, refPoints, maxMin, ...
316     weight)
317 % Function: calcWeightNormDistance(points, refPoints, maxMin, weight)
318 % Description: Calculate the weighted Euclidean distance from "points" to
319 % "refPoints"
320 % Parameters:
321 %   points(nPoint, N)       : each row is a point in N dimension space.
322 %   refPoints(nRefPoint, N) : each row is a reference point.
323 %   maxMin(1, N)            : normalized factor.
324 %   weight(1, N)            : weights
325 %
326 % Return:
327 %   distance(nPoint, nRefPoint)
328 %
329 %    Copyright 2011 by LSSSSWC
330 %    Revision: 1.0  Data: 2011-07-14
331 %*************************************************************************
332
333 nRefPoint = size(refPoints, 1);     % number of reference points
334 nPoint = size(points, 1);           % number of points
335
336 distance = zeros(nPoint, nRefPoint);
337 for ipt = 1:nRefPoint
338     refpt = refPoints(ipt, :);
339     for i = 1:nPoint
340         weightNormDist = ((points(i, :)-refpt) ./ maxMin).^2 .* weight;
341         distance(i, ipt) = sqrt(sum(weightNormDist));
342     end
343 end
344
345
346
347
348
349 function pop = calcCrowdingDistance(opt, pop, front)
350 % Function: pop = calcCrowdingDistance(opt, pop, front)
351 % Description: Calculate the 'crowding distance' used in the original
352 % NSGA-II.
```

```
% Syntax:
% Parameters:
% Return:
%
%    Copyright 2011 by LSSSSWC
%    Revision: 1.0  Data: 2011-07-11
%*************************************************************************

numObj = length( pop(1).obj );  % number of objectives
for fid = 1:length(front)
    idx = front(fid).f;
    frontPop = pop(idx);    % frontPop: individuals in front fid

    numInd = length(idx);   % nInd: number of individuals in current front

    obj = vertcat(frontPop.obj);
    obj = [obj, idx'];   % objctive values are sorted with individual ID
    for m = 1:numObj
        obj = sortrows(obj, m);

        colIdx = numObj+1;
        pop( obj(1, colIdx) ).distance = Inf;         % the first one
        pop( obj(numInd, colIdx) ).distance = Inf;    % the last one

        minobj = obj(1, m);         % the maximum of objective m
        maxobj = obj(numInd, m);    % the minimum of objective m

        for i = 2:(numInd-1)
            id = obj(i, colIdx);
            pop(id).distance = pop(id).distance + (obj(i+1, m)...
                - obj(i-1, m)) / (maxobj - minobj);
        end
    end
end
```

```
function result = nsga2(opt, nArchi, m, varargin)
% Function: result = nsga2(opt, varargin)
% Description: The main flowchart of of NSGA-II. Note:
%   All objectives must be minimization. If a objective is maximization,
%   the objective should be multipled by -1.
%
% Syntax:
%   result = nsga2(opt): 'opt' is generated by function nsgaopt().
%   result = nsga2(opt, param): 'param' can be any data type, it will be
%       pass to the objective function objfun().
%
```

```
12 %   Then ,the result structure can be pass to plotnsga to display the
13 %   population:  plotnsga(result);
14 %
15 % Parameters:
16 %   opt : A structure generated by funciton nsgaopt().
17 %   varargin : Additional parameter will be pass to the objective
18 %       functions. It can be any data type. For example, if you call:
19 %       nsga2(opt, param), then objfun would be called as objfun(x,param),
20 %       in which, x is the design variables vector.
21 % Return:
22 %   result : A structure contains optimization result.
23 %
24 %         LSSSSWC, NWPU
25 %   Revision: 1.2  Data: 2011-07-26
26 %*************************************************************************
27
28
29 tStart = tic();
30 %*************************************************************************
31 % Verify the optimization model
32 %*************************************************************************
33 opt = verifyOpt(opt);
34
35 %*************************************************************************
36 % variables initialization
37 %*************************************************************************
38 nVar    = opt.numVar;
39 nObj    = opt.numObj;
40 nCons   = opt.numCons;
41 nExtra  = opt.numExtra;
42 popsize = opt.popsize;
43
44 % pop : current population
45 % newpop : new population created by genetic algorithm operators
46 % combinepop = pop + newpop;
47 pop = repmat( struct(...
48     'var', zeros(1,nVar), ...
49     'obj', zeros(1,nObj), ...
50     'cons', zeros(1,nCons),...
51     'extra', zeros(1,nExtra),...
52     'rank', 0,...
53     'distance', 0,...
54     'prefDistance', 0,...       % preference distance used in R-NSGA-II
55     'nViol', 0,...
56     'violSum', 0),...
57     [1,popsize]);
58
59 % state: optimization state of one generation
60 state = struct(...
```

```
61 'currentGen', 1,...          % current generation number
62 'evaluateCount', 0,...       % number of objective function evaluation
63 'totalTime', 0,...           % total time from the beginning
64 'firstFrontCount', 0,...     % individual number of first front
65 'frontCount', 0,...          % number of front
66 'avgEvalTime', 0);           % average evaluation time of objective function
67 % (current generation)
68
69 % each row is the population of one generation
70 result.pops     = repmat(pop, [opt.maxGen, 1]);
71 % each row is the optimizaiton state of one generation
72 result.states   = repmat(state, [opt.maxGen, 1]);
73 % use for output
74 result.opt      = opt;
75
76 % global variables
77 global STOP_NSGA;   %STOP_NSGA : used in GUI , if STOP_NSGA~=0,
78 % then stop the optimizaiton
79 STOP_NSGA = 0;
80
81
82 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
83 % initialize the P0 population
84 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
85 ngen = 1;
86 pop = opt.initfun{1}(opt, pop, opt.initfun{2:end});
87 [pop, state] = evaluate(opt, pop, state, varargin{:});
88 [opt, pop] = ndsort(opt, pop);
89
90 % state
91 state.currentGen = ngen;
92 state.totalTime = toc(tStart);
93 state = statpop(pop, state);
94
95 result.pops(1, :) = pop;
96 result.states(1)  = state;
97
98 fileName = sprintf('optresult%d-M%d-G%d.txt',nArchi,m,ngen);
99 opt = output2file(opt, state, pop, 0, fileName, varargin);
100
101 % output
102 % plotnsga(result, ngen);
103 % opt = callOutputfuns(opt, state, pop);
104
105
106 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
107 % NSGA2 iteration
108 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
109 while( ngen < opt.maxGen && STOP_NSGA==0)
```

```
    % 0. Display some information
    ngen = ngen+1;
    state.currentGen = ngen;

    % 1. Create new population
    newpop = selectOp(opt, pop);
    newpop = crossoverOp(opt, newpop, state);
    newpop = mutationOp(opt, newpop, state);
    [newpop, state] = evaluate(opt, newpop, state, varargin{:});

    % 2. Combine the new population and old population:
    % combinepop = pop + newpop
    combinepop = [pop, newpop];

    % 3. Fast non dominated sort
    [opt, combinepop] = ndsort(opt, combinepop);

    % 4. Extract the next population
    pop = extractPop(opt, combinepop);

    % 5. Save current generation results
    state.totalTime = toc(tStart);
    state = statpop(pop, state);

    result.pops(ngen, :) = pop;
    result.states(ngen)  = state;


    fileName = sprintf('optresult%d-M%d-G%d.txt',nArchi,m,ngen);
    opt = output2file(opt, state, pop, 0, fileName, varargin);

end

% call output function for closing file
% opt = callOutputfuns(opt, state, pop, -1);
fileName = sprintf('optresult%d.txt',nArchi);
opt = output2file(opt, state, pop, 0, fileName, varargin);
fileName = sprintf('optresult%d-M%d.txt',nArchi,m);
opt = output2file(opt, state, pop, 0, fileName, varargin);


toc(tStart);
```

```
function defaultopt = nsgaopt()
% Function: defaultopt = nsgaopt()
% Description: Create NSGA-II default options structure.
```

```
% Syntax:  opt = nsgaopt()
%         LSSSSWC, NWPU
%   Revision: 1.3  Data: 2011-07-13
%*************************************************************************


defaultopt = struct(...
... % Optimization model
    'popsize', 50,...           % population size
    'maxGen', 100,...           % maximum generation
    'numVar', 0,...             % number of design variables
    'numObj', 0,...             % number of objectives
    'numCons', 0,...            % number of constraints
    'numExtra', 0,...           % number of extra vars
    'lb', [],...                % lower bound of design variables [1:numVar]
    'ub', [],...                % upper bound of design variables [1:numVar]
    'vartype', [],...           % variable data type [1:numVar]=real, 2=int
    'objfun', @objfun,...       % objective function
... % Optimization model components' name
    'nameObj',{{}},...
    'nameVar',{{}},...
    'nameCons',{{}},...
    'nameExtra',{{}},...
... % Initialization and output
... % population initialization function (use random number as default)
    'initfun', {{@initpop}},...
    'outputfuns',{{@output2file}},...   % output function
    'outputfile', 'populations.txt',... % output file name
    'outputInterval', 1,...             % interval of output
    'plotInterval', 5,...               % interval between two call of
                                        ...%"plotnsga".
... % Genetic algorithm operators
    ...% crossover operator (Ratio=1.2)
    'crossover', {{'intermediate', 1.2}},...
    ...% mutation operator (scale=0.1, shrink=0.5)
    'mutation', {{'gaussian',0.1, 0.5}},...
    ...% crossover fraction of variables of an individual
    'crossoverFraction', 'auto', ...
    ...% mutation fraction of variables of an individual
    'mutationFraction', 'auto',...
... % Algorithm parameters
    ...% compute objective function of a population in parallel.
    ...% {'yes','no'}
    'useParallel', 'no',...
    ...% number of workers use by parallel computation, 0 = auto select.
    'poolsize', 0,...
... % R-NSGA-II parameters
    ...% Reference point(s) used to specify preference. Each row is a
    ...%reference point.
```

```
    'refPoints', [],...
    ...% weight factor used in the calculation of Euclidean distance
    'refWeight', [],...
    ...% use normalized Euclidean distance by maximum and minumum
    ...% objectives possiable. {'front','ever','no'}
    'refUseNormDistance', 'front',...
    ...% parameter used in epsilon-based selection strategy
    'refEpsilon', 0.001 ...
);
```

```
function opt = output2file(opt, state, pop, type, fileName, varargin)
% Function: opt = output2file(opt, state, pop, type, varargin)
% Description: Output the population 'pop' to file. The file name is
% specified by 'opt.outputfile' field.
% Parameters:
%   type : output type.  -1 = the last call, close the opened file.
%          others(or no exist) = normal output
%   varargin : any parameter define in the options.outputfuns cell array.
%
%         LSSSSWC, NWPU
%    Revision: 1.2  Data: 2011-07-13
%*************************************************************************


% if(isempty(opt.outputfile))
    % return;  % the output file name is not specified, return directly
% end

% if( isfield(opt, 'outputfileFID') )
    % fid = opt.outputfileFID;
% else
    % fid = [];
% end

%*************************************************************************
% 1.Open the output file and output some population info
%*************************************************************************
% if( isempty(fid) )
    fid = fopen(fileName, 'w');
    if( fid == 0)
        error('NSGA2:OutputFileError',...
            'Can not open output file!! file name:%s', opt.outputfile);
    end
    opt.outputfileFID = fid;

    % Output some infomation
```

```
37      fprintf(fid, '#NSGA2\r\n');
38
39      fprintf(fid, 'popsize %d\r\n', opt.popsize);
40      fprintf(fid, 'maxGen %d\r\n', opt.maxGen);
41      fprintf(fid, 'numVar %d\r\n', opt.numVar);
42      fprintf(fid, 'numObj %d\r\n', opt.numObj);
43      fprintf(fid, 'numCons %d\r\n', opt.numCons);
44      fprintf(fid, 'numExtra %d\r\n', opt.numExtra);
45
46      % Output state field names
47      fprintf(fid, 'stateFieldNames\t');
48      names = fieldnames(state);
49      for i = 1:length(names)
50          fprintf(fid, '%s\t', names{i});
51      end
52      fprintf(fid, '\r\n');
53
54      fprintf(fid, '#end\r\n\r\n\r\n');
55  % end
56
57  %************************************************************************
58  % 2. If this is the last call, close the output file
59  %************************************************************************
60  if(type == -1)
61      fclose(fid);
62      rmfield(opt, 'outputfileFID');
63      return
64  end
65
66  %************************************************************************
67  % 3. Output population to file
68  %************************************************************************
69  fprintf(fid, '#Generation %d / %d\r\n', state.currentGen, opt.maxGen);
70
71  % output each state field
72  names = fieldnames(state);
73  for i = 1:length(names)
74      fprintf(fid, '%s\t%g\r\n', names{i}, getfield(state, names{i}));
75  end
76  fprintf(fid, '#end\r\n');
77
78  for i = 1:opt.numVar
79      fprintf(fid, 'Var%d\t', i);
80  end
81  for i = 1:opt.numObj
82      fprintf(fid, 'Obj%d\t', i);
83  end
84  for i = 1:opt.numCons
85      fprintf(fid, 'Cons%d\t', i);
```

```
end
for i = 1:opt.numExtra
    fprintf(fid, 'Extra%d\t', i);
end
fprintf(fid, '\r\n');

for p = 1 : opt.popsize
    for i = 1:opt.numVar
        fprintf(fid, '%g\t', pop(p).var(i) );
    end
    for i = 1:opt.numObj
        fprintf(fid, '%g\t', pop(p).obj(i) );
    end
    for i = 1:opt.numCons
        fprintf(fid, '%g\t', pop(p).cons(i));
    end
    for i = 1:opt.numExtra
        fprintf(fid, '%g\t', pop(p).extra(i));
    end
    fprintf(fid, '\r\n');
end

fprintf(fid, '\r\n\r\n\r\n');

fclose(fid);
```

```
function [] = PlotPareto(maxArchi,popsize,numObj, numExtra)
% Plot the global Pareto Frontier
% INPUTS:
% optresult files: text documents with the optimized points
% maxArchi: Number of Architectures

%% INITIALIZATION
nbObj = ones(1,maxArchi)*numObj;
nbExtra = ones(1,maxArchi)*numExtra;
nbDV = zeros(1,maxArchi);
nbCstn = zeros(1,maxArchi);
for j = 1:maxArchi
    [nbDV(j), nbCstn(j)] = Architecture(j);
end

%% SCAN FILES
for j = 1:maxArchi
    fid=fopen(sprintf('optresult%d.txt',j),'r');
    for i = 1:20
        fgetl(fid);
```

```
    end
    A=fscanf(fid,'%f',[nbDV(j)+nbObj(j)+nbCstn(j)+nbExtra(j) inf]).';
    V{j} = A;
    fclose(fid);
end

%% PARETO POINTS
optpt=[];
for j=1:maxArchi
    optpt = vertcat(optpt,V{j}(:,nbDV(j)+1:nbDV(j)+nbObj(j)));
    [~, lign] = prtp(optpt);
end

%% PLOT

figure
hold on

if nbObj(1) == 2;
    for j = 1:maxArchi
        scatter(optpt(lign(lign<=j*popsize & lign>popsize*(j-1)),1),...
            optpt(lign(lign<=j*popsize & lign>popsize*(j-1)),2),'x');
    end
elseif nbObj(1) == 3;
    for j = 1:maxArchi
        scatter3(optpt(lign(lign<=j*popsize & lign>popsize*(j-1)),1),...
            optpt(lign(lign<=j*popsize & lign>popsize*(j-1)),2),...
            optpt(lign(lign<=j*popsize & lign>popsize*(j-1)),3),'x');
    end
end

legend('Architecture 1','Architecture 2','Architecture 3',...
    'Architecture 4','Architecture 5')
end
```

```
function [A varargout]=prtp(B)
% Extract the pareto points from matrix B
A=[]; varargout{1}=[];
sz1=size(B,1);
jj=0; kk(sz1)=0;
c(sz1,size(B,2))=0;
bb=c;
for k=1:sz1
  j=0;
  ak=B(k,:);
  for i=1:sz1
```

```
    if i~=k
      j=j+1;
      bb(j,:)=ak-B(i,:);
    end
  end
  if any(bb(1:j,:)'<0)
    jj=jj+1;
    c(jj,:)=ak;
    kk(jj)=k;
  end
end
if jj
  A=c(1:jj,:);
  varargout{1}=kk(1:jj);
else
  warning('Points:Pareto',...
    'There are no Pareto points. The result is an empty matrix.')
end
```

```
function gen = Resultat(pcid)

gen = [5,0];
count = [0,1];
```

```
function newpop = selectOp(opt, pop)
% Function: newpop = selectOp(opt, pop)
% Description: Selection operator, use binary tournament selection.
%
%         LSSSSWC, NWPU
%    Revision: 1.1  Data: 2011-07-12
%*************************************************************************

popsize = length(pop);
pool = zeros(1, popsize);   % pool : the individual index selected

randnum = randi(popsize, [1, 2 * popsize]);

j = 1;
for i = 1:2:(2*popsize)
    p1 = randnum(i);
    p2 = randnum(i+1);

```

```
    if(~isempty(opt.refPoints))
        % Preference operator (R-NSGA-II)
        result = preferenceComp( pop(p1), pop(p2) );
    else
        % Crowded-comparison operator (NSGA-II)
        result = crowdingComp( pop(p1), pop(p2) );
    end

    if(result == 1)
        pool(j) = p1;
    else
        pool(j) = p2;
    end

    j = j + 1;
end
newpop = pop( pool );



function result = crowdingComp( guy1, guy2)
% Function: result = crowdingComp( guy1, guy2)
% Description: Crowding comparison operator.
% Return:
%   1 = guy1 is better than guy2
%   0 = other cases
%
%         LSSSSWC, NWPU
%    Revision: 1.0  Data: 2011-04-20
%*************************************************************************

if((guy1.rank < guy2.rank) || ((guy1.rank == guy2.rank) && ...
        (guy1.distance > guy2.distance) ))
    result = 1;
else
    result = 0;
end



function result = preferenceComp(guy1, guy2)
% Function: result = preferenceComp(guy1, guy2)
% Description: Preference operator used in R-NSGA-II
% Return:
%   1 = guy1 is better than guy2
%   0 = other cases
%
%    Copyright 2011 by LSSSSWC
%    Revision: 1.0  Data: 2011-07-11
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

if(  (guy1.rank  < guy2.rank) || ...
    ((guy1.rank == guy2.rank) && (guy1.prefDistance < guy2.prefDistance)) )
    result = 1;
else
    result = 0;
end
```

```
function state = statpop(pop, state)
% Function: state = statpop(pop, state)
% Description: Statistic Population.
%
%         LSSSSWC, NWPU
%    Revision: 1.0  Data: 2011-04-20
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%


N = length(pop);
rankVec = vertcat(pop.rank);
rankVec = sort(rankVec);

state.frontCount = rankVec(N);
state.firstFrontCount = length( find(rankVec==1) );
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Test Problem : 'CONSTR'
% Description:
%   (1)constrained
%
% Reference : [1] Deb K, Pratap A, Agarwal S, et al. A fast and elitist
%   multiobjective genetic algorithm NSGA-II[J]. Evolutionary Computation.
%   2002, 6(2): 182-197.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function [] = TP_CONSTR(nArchi, gen, ini, popsize, numObj, numExtra, ...
    useParallel,numProc,m)

%% CREATE DEFAULT OPTIONS STRUCTURE
options = nsgaopt();

%% INPUTS FROM MAIN FILE
options.popsize = popsize;
```

```
options.numObj = numObj;
options.numExtra = numExtra;
options.maxGen  = gen;                  % max generation
if ini == 0;
    % population initialization
    options.initfun = {@initpop,sprintf('optresult%d.txt',nArchi)};
end
options.useParallel = useParallel;
options.poolsize = numProc;

%% INPUTS FROM ARCHITECTURES
[...
    options.numVar,...                  % number of design variables
    options.numCons,...                 % number of constraints
    options.lb,...                      % lower bound of x
    options.ub,...                      % upper bound of x
    options.objfun,...
    options.vartype,...
    ] = Architecture(nArchi);

%% Optimization
result = nsga2(options,nArchi,m);               % begin the optimization!
```

```
function var = varlimit(var, lb, ub)
% Function: var = varlimit(var, lb, ub)
% Description: Limit the variables in [lb, ub].
%
%         LSSSSWC, NWPU
%    Revision: 1.0  Data: 2011-04-20
%*************************************************************************

numVar = length(var);
for i = 1:numVar
    if( var(i) < lb(i) )
        var(i) = lb(i);
    elseif( var(i) > ub(i) )
        var(i) = ub(i);
    end
end
```

```
function opt = verifyOpt(opt)
% Function: opt = verifyOpt(opt)
```

```
% Description: Verify the optimization model.
%         LSSSSWC, NWPU
%   Revision: 1.1  Data: 2011-07-15
%*********************************************************************


%*********************************************************************
%  popsize
%*********************************************************************
if ( mod(opt.popsize, 2) ~= 0 )
    warning('NSGA2:PopSizeError',...
        'The population size shoud be even number!%d => %d',...
        opt.popsize, opt.popsize+1);
    opt.popsize = opt.popsize + 1;
end

%*********************************************************************
% lb, ub
%*********************************************************************
if( length(opt.lb)~=opt.numVar || length(opt.lb)~=opt.numVar )
    error('NSGA2:OptModelError',...
        strcat('The numbers of lower and upper bounds(%d,%d)',...
        'should be equal to the design variable number(%d)!'), ...
        length(opt.ub), length(opt.lb), opt.numVar);
end

%*********************************************************************
% vartype
%*********************************************************************
if( length(opt.vartype) ~= opt.numVar )
    warning('NSGA2:OptModelWarning',...
    strcat('Design variables'' data type error! All the type',...
    ' is set to REAL coding (vartype=1)!'));
    opt.vartype = ones(1, opt.numVar);
end

%*********************************************************************
% nameObj, nameVar, nameCons
%*********************************************************************
if( ~iscell(opt.nameObj) || ~iscell(opt.nameVar) || ~iscell(opt.nameCons))
    error('NSGA2:OptModelError',...
        strcat('The names of objectives, design variables or',...
        'constraints should be specified in cell array,',...
        'for example, {''obj1'',''obj2''}'));
end

if( (~isempty(opt.nameObj)  && length(opt.nameObj)~=opt.numObj) || ...
    (~isempty(opt.nameVar)  && length(opt.nameVar)~=opt.numVar) || ...
    (~isempty(opt.nameCons) && length(opt.nameCons)~=opt.numCons))
```

```
52      error('NSGA2:OptModelError',...
53          strcat('All names of objectives, design variables or',...
54          ' constraints should be specified, if one is specified!'));
55  end
56
57  %*************************************************************************
58  % useparallel
59  %*************************************************************************
60  if( ~ischar(opt.useParallel) || ...
61      isempty( find(strcmpi(opt.useParallel, {'yes', 'no'}))) )
62      error('NSGA2:OptParamError', 'useParallel can be only "yes" or "no"!');
63  end
64
65  %*************************************************************************
66  % R-NSGA-II parameters
67  %*************************************************************************
68  % refPoints
69  if( ~isempty(opt.refPoints) && size(opt.refPoints,2)~=opt.numObj)
70      error('NSGA2:OptParamError',...
71          'The reference points has the format refPoints(nPoint, numObj)!');
72  end
73  % refWeight
74  if( ~isempty(opt.refPoints) && ~isempty(opt.refWeight) &&...
75          length(opt.refWeight)~=opt.numObj)
76      error('NSGA2:OptParamError',...
77      strcat('The weight factor vector used in R-NSGA-II must',...
78      'has the length of numObj!'));
79  end
```

## REFERENCES


[1] 3F, "MAP-H is a universal cost estimating model." Available online `http://3f-fr.com/products/existing-models/map-h?lang=en` accessed 07.11.2014, 2014.

[2] 4COST, "ACES is the most innovative parametric model available." Available online `http://www.4cost.co.uk/en/software/parametric-cost-estimating` accessed 07.11.2014, 2014.

[3] ABENSUR, T., "Innovative avionics and propulsion architectures for a "designed to safety" space vehicle," in *42nd AIAA/ASME/SAE/ASEE Joint Propulsion Conference & Exhibit*, 2006.

[4] ACTON, D. E. and OLDS, J. R., "Computational frameworks for collaborative multidisciplinary design of complex systems," in *7th AIAA/USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization*, 1998.

[5] ADVISORY GROUP FOR AEROSPACE RESEARCH AND DEVELOPMENT, "Propulsion and energy issues for the 21st century," tech. rep., North Atlantic Treaty Organization, 1997.

[6] AEROMOBIL, "Aeromobil - drive and fly." Available online `http://www.aeromobil.com/` accessed 10.14.2014, 2013.

[7] AGERWALA, T., "Putting petri nets to work," *IEEE Computer*, vol. 12, no. 12, pp. 85 – 94, 1979.

[8] AIR FORCE, "General requirements for safety engineering of systems and associated subsystems and equipment," tech. rep., The United States Air Force, 1963.

[9] AIRBUS, "Getting grips with aircraft performance," in *Airbus Industrie*, 2000.

[10] AKIN, D. L., "Mass estimating relationships - principles of space systems design." University Lecture, 2005.

[11] ALI, R. and AL-SHAMMA, O., "A comparative study of cost estimation models used for preliminary aircraft design," *Global Journal of Researches in Engineering*, vol. 14, no. 4, pp. 8 – 18, 2014.

[12] ALLISON, J. T., "Complex system optimization: A review of analytical target cascading, collaborative optimization, and other formulations," Master's thesis, School of Mechanical Engineering, University of Michigan, 2004.

[13] ANDERSON, J. D., *Aircraft Performance and Design*. Thomas Casson, 1999.

[14] ANDERSON, P. L., MCLELLAN, R. D., OVERTON, J. P., and WOLFRAM, G. L., *The Universal Tuition Tax Credit: A Proposal to Advance Parental Choice in Education*. Mackinac Center For Public Policy, 1997.

[15] ANSYS, Inc., "ANSYS: Computational Fluid Dynamics (CFD) software." Available online `http://ansys.com/Products/Simulation+Technology/Fluid+Dynamics` accessed 08.16.2014, 2014.

[16] APEX Admin, "APEX survey insights: What passengers get up to at 35,000 feet," tech. rep., Airline Passenger Experience Association, 2014.

[17] Ardema, M. D., "Advances in hypersonic vehicle synthesis with application to studies of advanced thermal protection systems," tech. rep., NASA Ames Research Center, 1995.

[18] Armstrong, M., de Tenorio, C., Garcia, E., and Mavris, D. N., "Function based architecture design space definition and exploration," in *26th Congress of International Council of the Aeronautical Sciences (ICAS)*, 2008.

[19] Arnold, C. R., Stone, R. B., and McAdams, D. A., "MEMIC: An interactive morphological matrix tool for automated concept generation," in *Proceedings of the 2008 Industrial Engineering Research Conference*, 2008.

[20] Ascani, L., "A Structural Weight Estimation Program (SWEEP) for aircraft," tech. rep., Rockwell International Corporation, 1974.

[21] Asselin, M., *An Introduction to Aircraft Performance*. AIAA Education Series, 1965.

[22] Associate Administrator for Commercial Space Transportation (AST), "Reusable launch vehicle programs and concepts," tech. rep., Federal Aviation Administration, 1998.

[23] ASTOS Solutions, "ASTOS, aerospace trajectory analysis tool for launch, reentry and orbit vehicles," tech. rep., ASTOS Solutions GmbH, 2008.

[24] Astrocentral, "The X Prize." Available online `http://www.astrocentral.co.uk/x-prize.html` accessed 04.21.2014, 2008.

[25] Balesdent, M., Berend, N., Depince, P., and Chriette, A., "Multidisciplinary design optimization of multi-stage launch vehicle using flight phases decomposition," *International Journal for Simulation and Multidisciplinary Design Optimization*, vol. 4, no. 3, pp. 117 – 125, 2010.

[26] Balkanyi, L. R. and Spurlock, O., "DUKSUP - a high thrust trajectory optimization code," in *AIAA/AHS/ASEE Aerospace Design Conference*, 1993.

[27] Bandte, O. and Mavris, D. N., "Multi-objective optimization using a joint probabilistic technique," in *8th AIAA/NASA/USAF/ISSMO Symposium on Multidisciplinary Analysis and Optimization*, 2000.

[28] Bandte, O., Mavris, D. N., and DeLaurentis, D. A., "Viable designs through a joint probabilistic estimation technique," in *4th World Aviation Congress and Exposition*, 1999.

[29] BBC, "1967: Russian cosmonaut dies in space crash." Available online `http://news.bbc.co.uk/onthisday/hi/dates/stories/april/24/newsid_2523000/2523019.stm` accessed 08.16.2015, 2014.

[30] BBC News, "The Astrium space jet." Available online http://news.bbc.co.uk/2/shared/spl/hi/pop_ups/07/sci_nat_the_astrium_space_jet/html/1.stm accessed 04.21.2014, 2007.

[31] Beard, S. and Starzyk, J., "Space tourism market study: suborbital space travel," tech. rep., Futron Corporation, 2002.

[32] Ben-Haim, Y., *Info-Gap Decision Theory*. Elsevier Ltd., 2006.

[33] Benson, D. A., Huntington, G. T., Thorvaldsen, T. P., and Raox, A. V., "Direct trajectory optimization and costate estimation via an orthogonal collocation method," *Journal of Guidance, Control and Dynamics*, vol. 29, no. 6, pp. 1435 – 1440, 2006.

[34] Berry, T., "Product and brand failures: a marketing perspective," 2010.

[35] Betts, J. T., "Survey of numerical methods for trajectory optimization," *Journal of Guidance, Control and Dynamics*, vol. 21, no. 2, pp. 193 – 207, 1998.

[36] Biondini, F., Bontempi, F., and Malerba, P. G., "Fuzzy reliability analysis of concrete structures," *Computers & Structures*, vol. 82, pp. 1033 – 1052, 2004.

[37] Blackwell, J., "Numerical method to calculate the induced drag or optimal span loading for arbitrary non-planar aircraft," tech. rep., National Aeronautics and Space Administration, 1976.

[38] Blau, P., "Launch vehicle library." Available online http://www.spaceflight101.com/launch-vehicle-library.html accessed 04.04.2015, 2011.

[39] Blieker, J. L., Garfinke, J. B., and Marks, K. E., "Development and production cost estimating relationships for aircraft turbine engines," tech. rep., RAND Corporation, 1982.

[40] Blue Origin, "Blue Origin - gallery." Available online https://www.blueorigin.com/gallery accessed 01.11.2016, 2014.

[41] Blue Origin, "New Shepard." Available online https://www.blueorigin.com/technology accessed 09.15.2015, 2015.

[42] Boggia, S. and Rud, K., "Intercooled recuperated aero engine," in *Advanced Project Design, MTU Aero Engines*, 2005.

[43] Boiffier, J.-L., *Dynamique du Vol de l'avion*. SupAéro - Département Aéronefs, 2001.

[44] Bonnet, A., *Mécanique des Fluides et Aérodynamique*. ISAE-Supaéro, 2010.

[45] Borer, N. K. and Mavris, D. N., "Requirements exploration for a notional multirole fighter," in *AIAA 3rd Annual Aviation Technology, Integration, and Operations (ATIO) Tech*, 2003.

[46] Bowcutt, K., Gonda, M., Hollowell, S., and Ralston, T., "Performance, operational and economic drivers of reusable launch vehicles," in *AIAA, 38th Joint Propulsion Conference*, 2002.

[47] Brady, J. F. and Lynch, R. A., "Reusable orbital transport second summary," tech. rep., National Aeronautics and Space Administration, 1965.

[48] Brauer, G., Cornick, D., and Stevenson, R., "Capabilities and applications of the Program to Optimize Simulated Trajectories (POST)," tech. rep., National Aeronautics and Space Administration, 1977.

[49] Braun, R. D., *Collaborative optimization: an architecture for large-scale distributed design*. PhD thesis, Stanford University, 1996.

[50] Braun, R. D., Moore, A., and Kroo, I. M., "Use of the collaborative optimization architecture for launch vehicle design," in *AIAA, NASA, and ISSMO, Symposium on Multidisciplinary Analysis and Optimization*, 1996.

[51] Braun, R. D., Moore, A. A., and Kroo, I. M., "Collaborative approach to launch vehicle design," *Journal of Spacecraft and Rockets*, vol. 34, no. 4, pp. 478 –486, 1997.

[52] Braun, R. D., Powell, R. W., Lepsch, R. A., and Stanley, D. O., "Comparison of two multidisciplinary optimization strategies for launch-vehicle design," *Journal of Spacecraft and Rockets*, vol. 32, no. 3, pp. 404 –410, 1995.

[53] Brooks, S. and Morgan, B., *Optimization using simulated annealing*. The Statistician, 1995.

[54] Brooks, S. P., "A hybrid optimization algorithm," *Royal Statistical Society*, vol. 44, no. 4, pp. 530 – 552, 1995.

[55] Brothers, B., "Launch vehicle mass estimating relationship database," tech. rep., National Aeronautics and Space Administration, 2000.

[56] Brown, L. D., Martin, M. J., Aleck, B. J., and Landes, R., "Composite-reinforced propellant tanks," tech. rep., National Aeronautics and Space Administration, 1975.

[57] Brundret, G., "Comfort and health in commercial aircraft: a literature review," *The Journal of The Royal Society for the Promotion of Health*, vol. 121, pp. 29 – 37, March 2001.

[58] Bruno, C. and Czysz, A., *Future Spacecraft Propulsion Systems*. Springer, 2009.

[59] Bruno, C., *Nuclear Propulsion*. InTech, 2012.

[60] Bryson, A. E., Desai, M. N., and Hoffman, W. C., "The energy-state approximation in performance optimization of supersonic aircraft," in *AlAA Guidance, Control, and High1 Dynamics conlerence*, 1968.

[61] Bryson, A. E., Desai, M. N., and Hoffman, W. C., "Energy-state approximation in performance optimization of supersonic aircraft," *Journal of Aircraft*, vol. 6, no. 6, pp. 481 – 488, 1969.

[62] Bucher, D. A. and Mavris, D. N., "Qualitative robust systems engineering analysis of apollo concepts and architectures," in *1st Space Exploration Conference: Continuing the Voyage of Discovery*, 2005.

[63] Buckley, M. J., Fertig, K. W., and Smith, D., "Design sheet: An environment for facilitating flexible trade studies during conceptual design," in *Aerospace Design Conference*, 1992.

[64] Buhl, W., Ebert, K., and Herbst, H., "Optimal ascent trajectories for advanced launch vehicles," in *AIAA, International Aerospace Planes Conference*, 1992.

[65] Buonanno, M. A. and Mavris, D. N., "Aerospace vehicle concept selection using parallel, variable fidelity genetic algorithms," in *10th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference*, 2004.

[66] Burgaud, F., Durand, J.-G., and Mavris, D. N., "A risk-aversion-based project valuation method to determine optimal technology infusion and architecture in aircraft design," in *17th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference*, 2016.

[67] Burgaud, F., Frank, C. P., and Mavris, D. N., "An aircraft development methodology aligning design and strategy to support key decision making," in *57th AIAA/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference*, 2016.

[68] Camcom, "Camcom espace: Les dossiers spatiaux." Available online `http://www.capcomespace.net/dossiers/index.htm` accessed 09.16.2015, 2015.

[69] Carapezza, E. M., "Planning sensing actions for UAVs in urban environments," in *Proceedings of SPIE*, vol. 5986, 2005.

[70] Cartagena, M. A., Rosario, J. E., and Mavris, D. N., "A method for technology identification, evaluation, and selection of aircraft propulsion systems," in *36th AIAA/ASME/SAE/ASEE Joint Propulsion Conference*, 2000.

[71] Cartier, A. and Cristoiu, I., "Space tourism: Regulatory framework of private initiatives and projects with a special interest on RLV regulations," in *International Astronautical Federation*, 2008.

[72] Castellini, F., *Multi-Disciplinary Design Optimization For Expendable Launch Vehicles*. PhD thesis, Politecnico Di Milano, Aerospace Engineering Department, 2012.

[73] Cavcar, M., "The International Standard Atmosphere (ISA)," tech. rep., Anadolu University, 2000.

[74] Centre for Atmospheric Science, "The Earth's atmosphere." Available online `http://www.atm.ch.cam.ac.uk/tour/atmosphere.html` accessed 07.26.2015, 1998.

[75] Centre National d'Etudes Spatiales, *The Middle Atmosphere and Space Observations*. Cepadues Editions, 1991.

[76] Chang, I.-S., "Investigation of space launch vehicle catastrophic failures," *Journal of Spacecraft and Rockets*, vol. 33, no. 2, pp. 198 – 205, 1996.

[77] Chavagnac, C. and Laporte-Weywada, H., "The suborbital space tourism project of EADS Astrium," in *45th AIAA/ASME/SAE/ASEE Joint Propulsion Conference & Exhibit*, 2009.

[78] Cheapflights.com, "The quest for comfort: Which airline has the most legroom?." Available online http://www.cheapflights.com/news/airline-legroom-guide/ accessed 07.14.2015, 2014.

[79] Chelouah, R. and Siarry, P., "A continuous genetic algorithm designed for the global optimization of multimodal functions," *Journal of Heuristics*, vol. 6, pp. 191 – 213, 2000.

[80] Chen, C.-H., "Robust design based on fuzzy optimization," *Tamsui Oxford Journal of Mathematical Sciences*, vol. 20, no. 1, pp. 65 – 72, 2004.

[81] Chen, Y.-S., Chou, T. H., and Wu, J. S., "Hybrid rocket propulsion technology for sounding rocket development," tech. rep., National Space Organization, 2008.

[82] Cheng, B. and Titterington, D. M., "Neural network: a review from a statistical perspective," *Statistical Science*, vol. 9, no. 1, pp. 2 – 54, 1994.

[83] Cheong, M.-P., Berleant, D., and Sheble, G., "Information gap decision theory as a tool for strategic bidding in competitive electricity markets," in *Probabilistic Methods Applied to Power Systems, 2004 International Conference on*, pp. 421–426, Sept 2004.

[84] Choi, H.-J., *A Robust Design Method for Model and Propagated Uncertainty*. PhD thesis, School of Mechanical Engineering, Georgia Institute of Technology, 2005.

[85] Chow, D., "NASA tests orion spaceship's parachutes with Mock Glitch." Available online http://www.space.com/21155-orion-parachute-test.html accessed 07.27.2015, 2013.

[86] Christensen, C. and Gresham, E., "Ten-year forecast of markets and launches for suborbital vehicles," in *AIAA SPACE 2012 Conference and Exposition*, 11-13 September 2012.

[87] Chung, H.-S., Choi, S., and Alonso, J., "Supersonic business jet design using knowledge-based genetic algorithm with adaptive, unstructured grid methodology," in *21st AIAA Applied Aerodynamics Conference*, 2003.

[88] Cleentano, J., Amorelli, D., Kohler, G., Adams, B., and Peters, P., "Nuclear propulsion in manned aerospace systems," in *1st AIAA Annual Meeting*, 1964.

[89] Cmglee, "Comparison U.S. standard atmosphere 1962." Available online http://en.wikipedia.org/wiki/U.S._Standard_Atmosphere#mediaviewer/File:Comparison_US_standard_atmosphere_1962.svg accessed 04.30.2014, 2014.

[90] Coello, C. A., Aguirre, A. H., and Zitzler, E., *Evolutionary Multi-Criterion Optimization*. Springer, 2005.

[91] Coello, C., Van-Veldhuizen, D. A., and Lamont, G. B., *Evolutionary Algorithms for Solving Multi-Objective Problems*. Springer Science & Business Media, 2002.

[92] Coinews Media Group, LLC, "Consumer price index data from 1913 to 2015." Available online http://www.usinflationcalculator.com/inflation/consumer-price-index-and-annual-percent-changes-from-1913-to-2008/ accessed 02.18.2015, 2015.

[93] Collange, G., Delattre, N., Hansen, N., Quinquis, I., and Schoenauer, M., *Multidisciplinary Design Optimization in Computational Mechanics*. John Wiley & Sons, Inc., 2013.

[94] Comparative Aircraft Flight Efficiency Foundation, "Why personal air vehicles?." Available online http://cafe.foundation/v2/pav_general_why.php accessed 03.29.2016, 2016.

[95] Conti, J. J., Holtberg, P. D., Diefenderfer, J. R., Napolitano, S. A., Schaal, A. M., Turnure, J. T., and Westfall, L. D., "Annual energy outlook 2014 with projections to 2040," tech. rep., U.S. Energy Information Administration, 2014.

[96] Convair Aerospace Division of General Dynamics, "Space Shuttle Synthesis Program (SSSP)," tech. rep., General Dynamics, 1970.

[97] Cook, E. L., "An exploratory investigation of weight estimation techniques for hypersonic flight vehicles," tech. rep., Wichita State University, 1981.

[98] Cramer, E. J., Dennis, J. E., Frank, P. D., Lewis, R. M., and Shubin, G. R., "Problem formulation for multidisciplinary optimization," *SIAM Journal on Optimization*, vol. 4, pp. 754 – 776, 1993.

[99] Creer, B. Y. and Smedal, H. A., "Centrifuge study of pilot tolerance to acceleration and the effects of acceleration on pilot performance," tech. rep., National Aeronautics and Space Administration, 1960.

[100] Croft, J., "Updated part 23 rules to revive four-seat GA market." Available online http://aviationweek.com/awin/updated-part-23-rules-revive-four-seat-ga-market accessed 10.15.2014, 2013.

[101] Crousse, G. L., "Design for passenger comfort and all weather operation," in *7th AIAA Aviation Technology, Integration and Operations Conference (ATIO)*, 2007.

[102] Damodaran, A., *Applied Corporate Finance*. John Wiley and Sons, Inc., 2011.

[103] Darlington, A., "CFD methods for aerodynamic design." Optimal Aerodynamics Ltd, 2014.

[104] Das, I. and Dennis, J. E., "Normal-boundary intersection: A new method for generating the Pareto surface in nonlinear multicriteria optimization problems," *SIAM Journal on Optimization*, vol. 8, no. 3, pp. 631 – 657, 1998.

[105] Dassault Systèmes, "Abaqus overview." Available online http://www.3ds.com/products-services/simulia/portfolio/abaqus/overview/ accessed 07.14.2014, 2014.

[106] Dassault Systèmes, "CATIA aerospace customers." Available online `http://www.3ds.com/customer-stories/all-customer-stories/` accessed 09.25.2015, 2015.

[107] Davenport, T. H. and Harris, J. G., "The prediction lover's handbook," tech. rep., MIT Sloan Management, 2009.

[108] David, L., "Rocketplane global overhauls suborbital craft." Available online `http://www.space.com/4557-rocketplane-global-overhauls-suborbital-craft.html` accessed 04.21.2014, 2007.

[109] De Baets, P. W. G. and Mavris, D. N., "Methodology for the parametric structural conceptual design of hypersonic vehicles," in *2000 World Aviation Conference*, 2000.

[110] Deb, K., Pretap, A., Agarwal, S., and Meyarivan, T., "A fast and elitist multiobjective genetic algorithm: NSGA-II," *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 2, pp. 182 – 197, 2002.

[111] Defense Logistics Agency, "Aerospace energy standard prices for DOD customers effective 1 oct 2014," tech. rep., Defense Logistics Agency, 2015.

[112] Defoort, S., Balesdent, M., Klotz, P., Schmollgruber, P., Morio, J., Hermetz, J., Blondeau, C., G.Carrier, and Bérend, N., "Multidisciplinary aerospace system design: Principles, issues and ONERA experience," *Journal Aerospace Lab*, vol. 4, pp. 1 – 15, 2012.

[113] Deore, P. J. and Patre, B. M., "Design of robust compensator for jet engine: An interval analysis approach," in *Control Applications, 2005. CCA 2005. Proceedings of 2005 IEEE Conference on*, 2005.

[114] Diehl, M., Bock, H. G., Diedam, H., and Wieber, P.-B., "Fast direct multiple shooting algorithms for optimal robot control," in *Fast Motions in Biomechanics and Robotics*, 2005.

[115] Dieter, G. and Schmidt, L., *Engineering Design*. Mc Graw Hill, 2013.

[116] Distefano, S. and Puliafito, A., "Dependability evaluation with dynamic reliability block diagrams and dynamic fault trees," *Dependable and Secure Computing, IEEE Transactions*, vol. 6, no. 1, pp. 4 – 17, 2009.

[117] Donahue, B., "Architecture selection: The key decision for human Mars mission planning," in *37th AIAA/ASME/SAE/ASEE Joint Propulsion Conference and Exhibit*, 2001.

[118] Dong, R., Sun, W., and Xu, H., "A novel non-probabilistic approach using interval analysis for robust design optimization," *Journal of Mechanical Science and Technology*, vol. 23, pp. 3199 –3208, 2009.

[119] Dong, R., Sun, W., and Xu, H., "Robust design optimization of gear parameters for a wind turbine using interval analysis," *Proceedings of the Institution of Mechanical Engineers Part C Journal of Mechanical Engineering Science*, vol. 224, pp. 2235 – 2245, 2010.

[120] Doran, E., Dyer, J., Lohner, K., Dunn, Z., Cantwell, B., and Zilliac, G., "Nitrous oxide hybrid rocket motor fuel regression rate characterization," in *43rd AIAA/ASME/SAE/ASEE Joint Propulsion Conference and Exhibit, Cincinnati, OH*, 2007.

[121] Doug Messier - Parabolic Arc, "NASA analysis: Falcon 9 much cheaper than traditional approach." Available online `http://www.parabolicarc.com/2011/05/31/nasa-analysis-falcon-9-cheaper-traditional-approach/` accessed 05.09.2014, 2011.

[122] Drela, M. and Youngren, H., "AVL overview." Available online `http://web.mit.edu/drela/Public/web/avl/` accessed 08.17.2014, 2014.

[123] Dreo, J., Petrowski, A., Siarry, P., and Taillard, E., *Metaheuristics for Hard Optimization: Methods and Case Studies*. Springer-Verlag Berlin Heidelberg, 2006.

[124] Du, L., Choi, K. K., and Lee, I., "Robust design concept in possibility theory and optimization for system with both random and fuzzy input variables," in *Proceedings of the ASME 2007 International Design Engineering Technical Conferences & Computers and Information in Engineering Conference*, 2007.

[125] Dubois, D. and Prade, H., "Operations on fuzzy numbers," *International Journal of Systems Science*, vol. 9, pp. 613 – 626, 1978.

[126] Dubois, D. and Prade, H., "Fuzzy real algebra: Some results," *Fuzzy Sets and Systems*, vol. 2, no. 4, pp. 327 – 348, 1979.

[127] Dugan, J. B., Bavuso, S. J., and Boyd, M. A., "Fault trees and markov models for reliability analysis of fault-tolerant digital systems," *Reliability Engineering and System Safety*, vol. 39, no. 3, pp. 291 – 307, 1993.

[128] EDS Technologies, "CATIA V5 composite part design-to-manufacturing process." Lecture, 2011.

[129] Elton, E. J., Gruber, M. J., Agrawal, D., and Mann, C., "Explaining the rate spread in corporate bonds," *Journal of Finance*, vol. 56, no. 1, pp. 247 – 277, 2001.

[130] Encyclopedia Astronautica, "Encyclopedia Astronautica: Black Armadillo." Available online `http://www.astronautix.com/craft/bladillo.htm` accessed 04.21.2014, 2014.

[131] Encyclopedia Astronautica, "Castor 120." Available online `http://www.astronautix.com/engines/casor120.htm` accessed 03.09.2015, 2015.

[132] Encyclopedia Astronautica, "GEM 40." Available online `http://www.astronautix.com/engines/gem40.htm` accessed 03.09.2015, 2015.

[133] Encyclopedia Astronautica, "Orion 50." Available online `http://www.astronautix.com/stages/orion50.htm` accessed 03.09.2015, 2015.

[134] Encyclopedia Astronautica, "Orion 50SP." Available online `http://www.astronautix.com/stages/orion50s.htm` accessed 03.09.2015, 2015.

[135] Encyclopedia Astronautica, "Voskhod KDU." Available online http://www.astronautix.com/craft/vosodkdu.htm accessed 07.27.2015, 2015.

[136] Engler, W. O., Biltgen, P. T., and Mavris, D. N., "Concept selection using an interactive reconfigurable matrix of alternatives (IRMA)," in *45th AIAA Aerospace Sciences Meeting and Exhibit*, 2007.

[137] Erfani, T. and Utyuzhnikov, S. V., "Handling uncertainty and finding robust Pareto frontier in multiobjective optimization using fuzzy," in *51st AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference*, 2010.

[138] Falkner, V. M., *The solution of lifting-plane problems by vortex-lattice theory: with seven appendices*. Aeronautical Research Council, 1953.

[139] Fan, A. and Bonnefoy, P., "Investigation of benefits and impacts of aircraft design cruise speed reductions on airlines operations and economics," in *AIAA Aviation Technology, Integration, and Operations Conference*, 2013.

[140] Far, B., "Dependability, reliability and testing of software systems." University Lecture, 2013.

[141] Farineau, T., Rabenasolo, B., Castelain, J. M., Meyer, Y., and Duverlie, P., "Use of parametric models in an economic evaluation step during the design phase," *The International Journal of Advanced Manufacturing Technology*, vol. 17, pp. 79 – 86, 2001.

[142] Faste, R. A., "The role of aesthetics in engineering," in *Japan Society of Mechanical Engineers (JSME) Journal*, 1995.

[143] Federal Aviation Administration, "Commercial Space Transportation." Available online http://www.faa.gov/regulations_policies/faa_regulations/commercial_space/ accessed 05.06.2014, 2013.

[144] Federal Aviation Administration, "Airworthiness certification of products and articles," tech. rep., U.S. Department of Transportation, 2015.

[145] Federal Aviation Administration, "Regulations and policies." Available online http://www.faa.gov/regulations_policies/ accessed 02.02.2016, 2015.

[146] Federal Aviation Administration, "Light-sport aircraft." Available online https://www.faa.gov/aircraft/gen_av/light_sport/ accessed 03.03.2016, 2016.

[147] Federal Reserve Bank of St. Louis, "Economic search - FRED economic data - BOAML yield spread," 2015.

[148] Finck, R. D., "USAF (United States Air Force) stability and control DATCOM (Data Compendium)," tech. rep., McDonnell Aircraft Co St Louis, 1977.

[149] Flittie, K. J., Jones, S., and Shaeffer, C., "HyFlyer: A hybrid propulsion suborbital launch vehicle," in *30th AIAA/ASME/SAE/ASEE Joint Propulsion Conference*, 1994.

[150] FlyFighterJet.com, "X-Cor Lynx space flight." Available online http://www.flyfighterjet.com/xcor-lynx-space-flight accessed 01.11.2016, 2015.

[151] Foncesca, C. M. and Fleming, P. J., "An overview of evolutionary algorithms in multiobjective optimization," *Journal of Evolutionary Computation*, vol. 3, no. 1, pp. 1 – 16, 1995.

[152] Forman, E. H. and Gass, S. I., "The analytic hierarchy process - an exposition," *Operations Research*, vol. 49, no. 4, pp. 469 – 486, 2001.

[153] Fossati, F. A. and Denaro, A., "A RLV concept selection from the thermo-mechanical outlook the lesson learned in FESTIP," in *9th International Space Planes and Hypersonic Systems and Technologies Conference and 3rd Weakly Ionized Gases Workshop*, 1999.

[154] Foust, J. and Smith, P., "Small launch vehicle services: Supply and demand through 2010," in *Space 2004 Conference and Exhibit*, 2004.

[155] Foust, J., "International suborbital safety proposal gets cold shoulder in U.S.." Available online http://spacenews.com/40615international-suborbital-safety-proposal-gets-cold-shoulder-in-us accessed 03.03.2016, 2014.

[156] Frank, C., Deveraux, M., Ausseil, R., and Mavris, D., "Design of an automated on-demand meal delivery system under emerging and evolving passenger requirements," in *57th AIAA Science and Technology Forum and Exposition 2016 (AIAA SciTech 2016)*, 2016.

[157] Frank, C., Durand, J.-G., Evain, H., Tyl, C., Mechentel, F., Brunet, A., and Lizy-Destrez, S., "Preliminary design of a new hybrid and technology innovative suborbital vehicle for space tourism," in *19th AIAA International Space Planes and Hypersonic Systems and Technologies*, 2014.

[158] Frank, C., Jimenez, H., Pfaender, H., and Mavris, D. N., "Scenario development to evaluate system-wide environmental benefits of aircraft technologies and concepts," in *AIAA Aviation Technology, Integration, and Operations Conference*, 2013.

[159] Frank, C. P. and Deveraux, M. N., "A consumer-centric methodology for selecting architectures against multiple objectives and its application to the in-flight catering and entertainment system," in *6th European Conference for Aerospace Sciences*, 2015.

[160] Frank, C. P., Durand, J.-G., Levy, A., Allair, F., and Mavris, D. N., "Design of an improved green taxiing system focused around the landing gear," in *14th AIAA Aviation Technology, Integration, and Operations Conference*, 2014.

[161] Frank, C. P., Levy, W. A., Durand, J.-G., Garcia, E., and Mavris, D. N., "An integrated and parametric environment for generation, selection and evaluation of new architectures at a conceptual level: Application to the environmental control system," in *AIAA SciTech - 52nd Aerospace Sciences Meeting*, 2014.

[162] Fuqua, N. B., "The applicability of markov analysis methods to reliability, maintainability, and safety," *Reliability Analysis Center*, vol. 10, no. 2, pp. 1 – 8, 2003.

[163] Futron Corporation, "Suborbital space tourism demand revisited," tech. rep., Futron Corporation, 2006.

[164] Gadd, K., *TRIZ for Engineers - Enabling Inventive Problem Solving*. John Wiley and Sons, Inc., 2011.

[165] Gallo, A., "A refresher on net present value," tech. rep., Harvard Business Review, 2014.

[166] Galorath Inc., "SEER project management tool overview." Available online `http://www.galorath.com/index.php/products` accessed 07.10.2014, 2014.

[167] Galvez, A. and Naja-Corbin, G., "ESA's view on private suborbital spaceflights," tech. rep., European Space Agency, 2008.

[168] Gehman, H. W., Barry, J. L., Deal, D. W., Hallock, N., Hess, K. W., Hubbard, G. S., Logson, J. M., Osheroff, D. D., Ride, S. T., Tetrault, R. E., Turcotte, S. A., Wallace, S. B., and Widnall, S. E., "Columbia - accident investigation board," tech. rep., Columbia Accident Investigation Board, 2003.

[169] Gerend, R. P. and Roundhill, J. P., "Correlation of gas turbine engine weights and dimensions," in *AIAA 6th Propulsion Joint Specialsit Conference*, 1970.

[170] German, B., "Multi-objective optimization: Principles and algorithms." University Lecture, 2013.

[171] German, B., "Multidisciplinary Design Optimization (MDO): single-level and multi-level methods." University Lecture, 2013.

[172] German, B., "Robust design." University Lecture, 2013.

[173] Ghosh, A. and Dehuri, S., "Evolutionary algorithms for multi-criterion optimization: A survey," *International Journal of Computing & Information Sciences*, vol. 2, no. 1, pp. 38 – 57, 2004.

[174] Glatt, C. R., "WAATS - a computer program for weights analysis of advanced transportation systems," tech. rep., National Aeronautics and Space Administration, 1974.

[175] Goeble, M. and Gruenwald, L., "A survey of data mining and knowledge discovery software tools," *SIGKDD Explorations*, vol. 1, no. 1, pp. 20 – 33, 1999.

[176] Goehlich, R. A., *Space Tourism: Economic and Technical Evaluation of Suborbital SPACE Flight for Tourism*. Der Andere Verlag, 2002.

[177] Gong, C., Chen, B., and Gu, L., "Design and optimization of RBCC powered suborbital reusable launch vehicle," in *19th AIAA International Space Planes and Hypersonic Systems and Technologies Conference*, 2014.

[178] Gonzales, D., Eisman, M., Shipbaugh, C., Bonds, T. M., and Le, A. T., "Proceedings of the RAND project air force workshop on transatmospheric vehicles," tech. rep., RAND Corporation, 1995.

[179] Gorgen, J. and Knab, O., "CryoROC - a multi-phase navier-stokes solver for advanced rocket thrust chamber design," *The SAO/NASA Astrophysics Data System*, vol. 487, pp. 631 – 638, 2002.

[180] Gorove, K., "Delimitation of outer space and the aerospace object - where is the law?," *Journal of Space Law*, vol. 28, no. 1, pp. 11 – 28, 1997.

[181] Gorove, S., "Aerospace object - legal and policy issues for air and space law," *Journal of Space Law*, vol. 25, no. 2, pp. 101 – 112, 1997.

[182] Graver, C. A. and Morrison, D. C., "CAC(Q) CERs for solid rocket motors," in *Annual Department of Defense Cost Analysis Symposium Paper*, 1993.

[183] Grigoriev, D., Harrison, J., and Hirsch, E. A., *Computer Science - Theory and Applications*. Springer, 2006.

[184] Grosse, M., "Development work on a small experimental hybrid rocket," in *33rd AIAA/ASME/SAE/ASEE, Joint Propulsion Conference & Exhibit*, 1997.

[185] Gulledge, T. R. and Litteral, L. A., *Cost Analysis Applications of Economics and Operations Research*. Springer-Verlag, 1989.

[186] Gunston, B., "Military jet engines." Available online `http://engines.fighter-planes.com/jet_engine.htm` accessed 04.01.2014, 2007.

[187] Gur, O., Mason, W. H., and Schetz, J. A., "Full-configuration drag estimation," *Journal of Aircraft*, vol. 47, no. 4, pp. 1356 – 1367, 2010.

[188] Guthrie, P., Christensen, C., Hay, J., Graham, R., DiBello, F., and Odyssey, A., "Suborbital space market identification and classification," in *AIAA Space 2011 Conference and Exposition*, 2011.

[189] H. Apgar, D. B., *Space Mission Analysis and Design*. Space Technology Library, 1999.

[190] Haigneré, J.-P., Gathier, L., and Coué, P., "Vehra-SH suborbital manned vehicle," in *57th International Astronautical Congress*, 2006.

[191] Haimes, Y. Y., Lasdon, L. S., and Wismer, D. A., "On a bicriterion formulation of the problems of integrated system identification and system optimization," *IEEE Transactions of Systems, Man, and Cybernetics*, vol. 1, pp. 296 – 297, 1971.

[192] Hall, R. and Shayler, D., *Soyuz: A Universal Spacecraft*. Springer Praxis, 2004.

[193] Hargraves, C. R., Paris, S. W., and Vlases, W. G., "OTIS past, present, and future," in *Astrodynamics Conference*, 1992.

[194] Harloff, G. J. and Berkowitz, B. M., "HASA - Hypersonic Aerospace Sizing Analysis for the preliminary design of aerospace vehicles," tech. rep., National Aeronautics and Space Administration, 1988.

[195] Harris, R. V., "An analysis and correlation of aircraft wave drag," tech. rep., National Aeronautics and Space Administration, 1964.

[196] Harris, R. V., "A numerical technique for analysis of wave drag at lifting condition," tech. rep., National Aeronautics and Space Administration, 1966.

[197] Heer, J. and Agrawala, M., "Design considerations for collaborative visual analytics," *Information Visualization*, vol. 7, pp. 49 – 62, 2008.

[198] Helmer, O., "Analysis of the future: The Delphi method," tech. rep., RAND Corporation, 1967.

[199] Heppenheimer, T. A., *History of the Space Shuttle*. Smithsonian Institution Press, 2002.

[200] Hermann, J. A. and Schmidt, D. K., "Fuel-optimal SSTO mission analysis of a generic hypersonic vehicle," in *Guidance, Navigation, and Control Conference*, 1995.

[201] Hess, R. W. and Romanoff, H. P., "Aircraft airframe cost estimating relationships: All mission types," tech. rep., RAND Corporation, 1987.

[202] Heuston Consulting, Inc, "The database of cost references by group," tech. rep., Heuston Consulting, Inc, 2009.

[203] Hipel, K. and Ben-Haim, Y., "Decision making in an uncertain world: information-gap modeling in water resources management," *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, vol. 29, pp. 506–517, Nov 1999.

[204] Hirschel, E. H. and Weiland, C., *Multidisciplinary Design Aspects*. Springer Berlin Heidelberg, 2009.

[205] Hoerner, S. F., *Fluid-Dynamic Drag, Theoretical, Experimental and Statistical Information*. Hoerner Fluid Dynamics, 1965.

[206] Hoerner, S. F. and Borst, H. V., *Fluid-Dynamic Lift*. Hoerner Fluid Dynamics, 1975.

[207] Hollingsworth, P. and Mavris, D. N., "A method for concept exploration of hypersonic vehicles in the presence of open & evolving requirements," in *2000 World Aviation Conference*, 2000.

[208] Horbach, J., Rammer, C., and Renningsber, K., "Determinants of eco-innovations by type of environmental impact-the role of regulatory push/pull, technology push and market pull," *Ecological economics*, vol. 78, pp. 112–122, June 2012.

[209] Howe, D., *Aircraft Conceptual Design Synthesis*. London: Professional Engineering Publishing, 2000.

[210] Hsu, J. C., Raghunathan, S., and Curran, R., "A proposed systems engineering diagnostic method," in *47th AIAA Aerospace Sciences Meeting Including The New Horizons Forum and Aerospace Exposition*, 2009.

[211] Huang, D. H. and Huzel, D. K., *Modern Engineering for Design of Liquid-Propellant Rocket Engines*. American Institute of Aeronautics and Astronautics, 1992.

[212] Huang, X. and Chuboda, B., "Application of a HTHL SAV design methodology to suborbital tourism vehicle," in *45th AIAA Aerospace Sciences Meeting and Exhibit*, 2007.

[213] Huang, X., Chuboda, B., and Coleman, G., "A generic hands-on conceptual design methodology applied to a tourist space access vehicle," in *44th AIAA Aerospace Sciences Meeting and Exhibit*, 2006.

[214] Huang, X. and Chudoba, B., "A trajectory synthesis simulation program for the conceptual design of a suborbital tourism vehicle," in *AIAA Modeling and Simulation Technologies Conference and Exhibit*, 2005.

[215] Huang, Z., Fint, J. A., and Kuck, F. M., "Key reliability drivers of liquid propulsion engines and a reliability model for sensitivity analysis," in *41st AIAA/ASME/SAE/ASEE Joint Propulsion Conference & Exhibit*, 2005.

[216] Humble, R. and Henry, G., *Space Propulsion Analysis and Design*. Space Technology Series, 1995.

[217] Hurschel, P. and Cox, T., "Launch condition deviations of reusable launch vehicle simulations in exo-atmospheric zoom climbs," in *AIAA Atmospheric Flight Mechanics Conference and Exhibit*, 2003.

[218] IATA, "2013 IATA global passenger survey highlights," tech. rep., IATA, 2013.

[219] Ishiguro, T., Sinohara, K., Sakio, K., and Nakagawa, I., "A study on combustion efficiency of paraffin-based hybrid rockets," in *47th AIAA/ASME/SAE/ASEE Joint Propulsion Conference & Exhibit 31 July - 03 August 2011, San Diego, California*, 2011.

[220] Jackson, A., "Preliminary design weight estimation program," tech. rep., Aero-Commander Div., 1971.

[221] Jimenez, H. and Mavris, D. N., "An evolution of morphological analysis applications in systems engineering," in *48th AIAA Aerospace Sciences Meeting Including the New Horizons Forum and Aerospace Exposition*, 2010.

[222] Jobe, C. E., *Thrust and Drag: Its Prediction and Verification*, vol. 98. Progress in Astronautics and Aeronautics, 1985.

[223] Johnsonbaugh, R. and Murata, T., "Petri nets and marked graphs – mathematical models of concurrent computation," *The American Mathematical Monthly*, vol. 89, no. 8, pp. 552 – 566, 1982.

[224] Johson, D., "The cost of certification." Available online `http://generalaviationnews.com/2012/09/09/the-cost-of-certification/` accessed 10.15.2014, 2012.

[225] Jones, R. T., "Theory of wing-body drag at supersonic speeds," tech. rep., National Advisory Committee for Aeronautics, 1956.

[226] Jumper, E. J., "Wave drag prediction using a simplified supersonic area rule," *Journal of Aircraft*, vol. 20, no. 10, pp. 893 – 895, 1983.

[227] Kalogeropoulos, S., "Sky rage," in *Flight Safety Australia*, pp. 36 – 37, July 1998.

[228] Kamdar, N., Smith, M., Thomas, R., Wikler, J., and Mavris, D. N., "Response surface utilization in the exploration of a supersonic business jet concept with application of emerging technologies," in *SAE 2003 Transactions Journal of Aerospace*, 2003.

[229] Karabeyoglu, A., "Hybrid rocket propulsion for future space launch." University Lecture, 2008.

[230] Karabeyoglu, M., Altman, D., and Cantwell, B., "Combustion of liquefying hybrid propellants: Part 1, general theory," *Journal of Propulsion and Power*, vol. 18, no. 3, pp. 610 – 620, 2002.

[231] Kasik, D. J., Elbert, D., Lebanon, G., Park, H., and Pottenger, W. M., "Data transformations and representations for computation and visualization," *Information Visualization: Special Issue on the Fundations and Frontiers of the Visual Analytics*, vol. 8, no. 4, pp. 275 – 285, 2009.

[232] Keim, D. A., Mansmannand, F., Schneidewind, J., Thomas, J., and Ziegler, H., *Visual Analytics: Scope and Challenges*. Springer-Verlag Berlin, Heidelberg, 2008.

[233] Keller, L. R. and Ho, J. L., "Decision problem structuring: Generating options," *IEEE Transactions of Systems, Man, and Cybernetics*, vol. 18, no. 5, pp. 715 – 728, 1988.

[234] Killian, J. R., Russell, B., Cooper, R. S., and Larrabee, E., "Bulletin of the atomic scientists," *Science and Public Affairs*, vol. 17, pp. 22 – 28, March 1962.

[235] Kinney, D. J., Bowles, J. V., Yang, L. H., and Roberts, C. D., "Conceptual design of a SHARP-CTV," in *35th AIAA Thermophysics Conference*, 2001.

[236] Kirby, M. R., *A Methodology for Technology Identification, Evaluation, and Selection in Conceptual and Preliminary Aircraft Design*. PhD thesis, School of Aerospace Engineering, Georgia Institute of Technology, 2001.

[237] Kiselev, A. I., Medvedev, A. A., and Menshikov, V. A., *Astronautics: Summary and Prospects*. Springer-Verlag Wien GmbH, 2001.

[238] Kizer, J. R. and Mavris, D. N., "Set-based design space exploration enabled by dynamic constraint analysis," in *29th Congress of the International Council of the Aeronautical Sciences*, 2014.

[239] Koelle, D. E., "The transcost-model for launch vehicle cost estimation and its application to future systems analysis," *Acta Astronautica*, vol. 11, no. 12, pp. 803 – 817, 1984.

[240] Koelle, D. E., *Handbook of Cost Engineering and Design of Space Transportation Systems*. TransCostSystems, 2013.

[241] Kokpitaero, "How to design an aircraft cabin." Available online `https://www.youtube.com/watchv=E0BuXu1OWw0` accessed 02.02.2015, 2011.

[242] Kothari, A. P. and Webber, D., "Potential demand for orbital space tourism opportunities made available via reusable rocket and hypersonic architectures," in *AIAA SPACE 2010 Conference & Exposition*, 2010.

[243] Kreim, H., Kugelmann, B., Pesch, H. J., and Breitner, M. H., "Minimizing the maximum heating of a re-entering Space Shuttle: An optimal control problem with multiple control constraints," *Optimal Control Applications and Methods*, vol. 17, pp. 45 – 69, 1996.

[244] Krevor, Z. C., *A Methodology to Link Cost and Reliability for Launch Vehicle Design*. PhD thesis, School of Aerospace Engineering, Georgia Institute of Technology, 2007.

[245] Krishnamachari, R., "Designing with fuzzy compromise decision support problems," Master's thesis, Department of Mechanical Engineering, University of Houston, 1991.

[246] Ku, F., "Economic analysis in tourism industry." University Lecture, 2012.

[247] Lai, Y. J., Liu, T. Y., and Hwang, C. L., "TOPSIS for MODM," *European Journal of Operational Research*, vol. 76, pp. 486 – 500, 1994.

[248] Lara, M. R., "ATK space propulsion products catalog," May 2008. Approved for Public Release OSR No. 08-S-0259 and OSR No. 08-S-1556; Dated 14 May 2008; Export Authority ITAR 125.4(b)(13).

[249] Larsson, E., "Simulation of flow and combustion in h2/o2 rocket thrust chambers using a 2D spray combustion method," Master's thesis, Chalmers University of Technology, 2012.

[250] Le Goff, T. and Moreau, A., "Astrium suborbital spaceplane project: Demand analysis of suborbital space tourism," *Acta Astronautica*, vol. 92, pp. 144 – 149, 2013.

[251] Lee, W. S., Grosh, D. L., Tillman, F. A., and Lie, C. H., "Fault tree analysis, methods, and applications - a review," *IEEE Transactions on Reliability*, vol. 34, no. 3, pp. 194 – 203, 1985.

[252] Lehot, F., Clervoy, J.-F., Caquelard, F., Cauchois, R., Coue, P., Dodelin, G., Gai, F., Gucciardi, R., Lefevre, C., Mora, C., and Rosier, P., *Embarquer dès Demain pour l'Espace*. Vuibert, 2010.

[253] LeVine, M. J., Kirby, M., and Mavris, D. N., "Noise-sensitivity to vehicle-level design variables," in *12th AIAA Aviation Technology, Integration, and Operations (ATIO) Conference and 14th AIAA/ISSM*, 2012.

[254] Lewe, J.-H., Laughlin, T., and Mavris, D. N., "Impacts of design characteristics on market demand of vertical takeoff and landing personal air vehicles," in *10th AIAA Aviation Technology, Integration, and Operations (ATIO) Conference*, 2010.

[255] Lewe, J.-H., Meyer, K. A., Unton, T. J., and Mavris, D. N., "Sensitivity analysis of consumer preference for a future small aircraft with point-to-point operations," in *9th AIAA Aviation Technology, Integration, and Operations Conference (ATIO)*, 2009.

[256] Lewis, B., "Complete 1976 standard atmosphere." Available online http://www.mathworks.com/matlabcentral/fileexchange/13635-complete-1976-standard-atmosphere accessed 07.12.2014, 2014.

[257] Liebeck, R. H., Andrastek, D. A., Chau, J., Girvin, R., Lyon, R., Rawdon, B. K., Scott, P. W., and Wright, R. A., "Advanced subsonic airplane design and economic studies," tech. rep., National Aeronautics and Space Administration, 1995.

[258] Lim, D., Justin, C., and Mavris, D., "Advanced general aviation concept study for a roadable aircraft," in *AIAA Aviation*, 2015.

[259] Linsey, J. S. and Becker, B., *Design Creativity 2010*. Springer London, 2011.

[260] Lintner, J., "The valuation of risk assets and the selection of risky investments in stock portfolios and capital budgets," *The Review of Economics and Statistics*, vol. 47, pp. 13 – 37, February 1965.

[261] Liu, J., "Weakest link theory and multiaxial criteria," in *5th International Conference on Biaxial/Multiaxial Fatigue and Fracture*, 1997.

[262] Lolis, P., "Preliminary aero engine weight estimation techniques for a techno-economic and environmental risk analysis framework (tera)." Lecture, 2012.

[263] Lomax, H., "The wave drag of arbitrary configurations in linearized flow as determined by area and forces in oblique planes," tech. rep., National Advisory Committee for Aeronautics, 1955.

[264] Lundgren, E. C. and Hanson, J. P., "Design and maintainability considerations regarding the effects of suborbital flights on composite constructed vehicles," in *53rd AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics and Materials Conference*, 2012.

[265] Luntz, S., "Are flying cars finally becoming reality?." Available online http://www.iflscience.com/technology/are-flying-cars-finally-becoming-reality accessed 08.17.2015, 2014.

[266] MacConochie, I. O. and Klich, P. J., "Techniques for the determination of mass properties of earth-to-orbit transportation systems," tech. rep., National Aeronautics and Space Administration, 1978.

[267] Marano, G. C. and Quaranta, G., "Fuzzy-based robust structural optimization," *International Journal of Solids and Structures*, vol. 45, pp. 3544 – 3557, 2008.

[268] Martin, J. C. and Law, G. W., "Suborbital reusable launch vehicles and applicable markets," tech. rep., The Aerospace Corporation: Space Launch Support Division - Space Launch Operations, 2002.

[269] Martin, J. D., "Robust kriging models," in *51st AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference*, 2010.

[270] Marx, W. J., Mavris, D. N., and Schrage, D. P., "Integrated product development for the wing structural design of the high speed civil transport," in *5th Symposium on Multidisciplinary Analysis and Optimization*, 1994.

[271] Mattingly, J. D., Heiser, W. H., and Pratt, D. T., *Aircraft Engine Design - Second Edition*. AIAA Education Series, 2002.

[272] Mavris, D. N., Bandte, O., and DeLaurentis, D. A., "Robust design simulation: A probabilistic approach to multidisciplinary design," *Journal of Aircraft*, vol. 36, no. 1, pp. 298 – 307, 1999.

[273] Mavris, D. N., de Tenorio, C., and Armstrong, M., "Methodology for aircraft system architecture definition," in *46th AIAA Aerospace Sciences Meeting and Exhibit*, 2008.

[274] Mavris, D. N. and DeLaurentis, D. A., "An integrated approach to military aircraft selection and concept evaluation," in *1st AIAA Aircraft Engineering, Technology, and Operations Congress*, 1995.

[275] Mavris, D. N., DeLaurentis, D. A., Bandte, O., and Hale, M. A., "A stochastic approach to multi-disciplinary aircraft analysis and design," in *36th Aerospace Sciences Meeting & Exhibit*, 1998.

[276] Mavris, D. N. and Kirby, M. R., "Technology identification, evaluation, and selection for commercial transport aircraft," in *Society of Allied Weight Engineers*, 1999.

[277] Mavris, D. N., Laurentis, D. A., Bandte, O., and Hale, M. A., "A stochastic approach to multi-disciplinary aircraft analysis and design," in *36th Aersopace Sciences Meeting & Exhibit*, 1998.

[278] Mavris, D. N. and Pinon, O. J., *An Overview of Design Challenges and Methods in Aerospace Engineering*. Scientific Publishing Services Pvt. Ltd, 2011.

[279] Mavris, D. N., Pinon, O. J., and Fullmer, D., "Systems design and modeling: A visual analytics approach," in *27th International Congress of the Aeronautical Sciences*, 2010.

[280] Mavris, D. N., Soban, D. S., and Largent, M. C., "An application of a technology impact forecasting (TIF) method to an uninhabited combat aerial vehicle," in *4th World Aviation Congress and Exposition*, 1999.

[281] Mavris, D. N., "Enabling technologies for strategic decision making of advanced design concepts." University Lecture, 2012.

[282] Mavris, D. N., "Introduction to decision making and multi-attribute decision making (MADM)." University Lecture, 2012.

[283] Mavris, D. N., "Mission adaptation & technology trends in aero propulsion." University Lecture, 2012.

[284] Mavris, D. N., "Technical feasibility and economic viability gap analysis." University Lecture, 2012.

[285] Mavris, D. N., "Introduction to probability and statistics - part I: Probability." University Lecture, 2013.

[286] McAfee, J., Culver, G., and Naderi, M., "NASA Air Force Cost Model (NAFCOM): Capabilities and results," in *2011 JANNAF MSS / LPS / SPS Joint Meeting*, 2011.

[287] McCormick, B. W., *Aerodynamics, Aeronautics and Flight Mechanics*. Wiley, 1995.

[288] McKay, M. D., Conover, W. J., and Beckam, R. J., "A comparison of three methods for selecting values of input variables in the analysis of output from a computer code," *Technometrics*, vol. 221, pp. 239 – 245, 1979.

[289] McManus, H. L., Hastings, D. E., and Warmkessel, J. M., "New methods for rapid architecture selection and conceptual design," *Journal of Spacecraft and Rockets*, vol. 41, pp. 10 – 19, 2004.

[290] Meder, D. S. and McLaughlin, J. R., "A generalized trajectory simulation system," in *Summer Computer Simulation Conference*, 1976.

[291] Meisl, C. J., "Life-cycle-cost considerations for launch vehicle liquid propellant rocket engine," *Journal of Propulsion*, vol. 4, no. 2, pp. 118 – 126, 1988.

[292] Meisl, C. J., "Rocket engine vs. jet engine comparison," in *AIAA/SAE/ASME/ASEE 28th Joint Propulsion Conference and Exhibit*, 1992.

[293] Messac, A. and Mattson, C. A., "Normal constraint method with guarantee of even representation of complete Pareto frontier," *AIAA Journal*, vol. 42, pp. 2101 – 2111, 2004.

[294] Meyer, M. L., Chato, D. J., Plachta, D. W., Zimmerli, G. A., Barsi, S. J., Dresar, N. T. V., and Moder, J. P., "Mastering cryogenic propellants," *Journal of Aerospace Engineering*, vol. 26, no. Special Issue: Seventy Years of Aerospace Research and Technology Excellence at NASA Glenn Research Center, pp. 343 – 351, 2013.

[295] Michalewicz, Z. and Arabas, J., "Genetic algorithms for the 0/1 knapsack problem," in *Proceedings of the 8th International Symposium on Methodologies for Intelligent Systems*, ISMIS '94, (London, UK), pp. 134 – 143, Springer-Verlag, 1994.

[296] Miele, A., *Flight Mechanics Volume I: Theory of Flight Paths*. Addison-Wesley, 1962.

[297] Modarres, M., Kaminskiy, M., and Krivtsov, V., *Reliability Engineering and Risk Analysis: A Practical Guide*. Library of Congress Cataloging-in-Publication Data, 1999.

[298] Montgomerie, B., "Design of a turbofan engine cycle with afterburner for a conceptual UAV," tech. rep., FOI - Swedish Defence Research Agency, 2005.

[299] Moody's Investor Service, "Rating symbols and definitions," tech. rep., Moody's Corporation, August 2015.

[300] Mora, N., Heotzman, N., Scoville, S., and Takahashi, T., "Conceptual design of a N+1 transonic executive jet," in *52nd Aerospace Sciences Meeting*, 2014.

[301] Mouawad, J. and White, M., "On jammed jets, sardines turn on one another." Available online http://www.nytimes.com/2013/12/23/business/on-jammed-jets-sardines-turn-on-one-another.html?smid=tw-share&_r=1 accessed 07.14.2015, 2013.

[302] MSC Software Corporation, "MSC Nastran - multidisciplinary structural analysis." Available online http://www.mscsoftware.com/product/msc-nastran accessed 07.14.2014, 2014.

[303] MSC Software Corporation, "Patran complete FEA modeling solution." Available online http://www.mscsoftware.com/product/patran accessed 07.14.2014, 2014.

[304] Murata, T., "Petri nets: Properties, analysis and applications," *Proceedings of the IEEE*, vol. 77, no. 4, pp. 541 – 580, 1989.

[305] Musgrave, G., Larsen, A., and Sgobba, T., *Safety Design for Space System*. Elsevier, 2009.

[306] Myers, R. H. and Montgomery, D. C., *Response Surface Methodology: Process and Product Optimization Using Designed Experiments*. Wiley Series in Probability and Statistics, 2002.

[307] Nam, T., "Introduction to FLOPS modeling." University Lecture, 2012.

[308] NASA, "NASA Air Force Cost Model - NAFCOM." Available online http://www.nasa.gov/offices/ooe/NAFCOM.html#.U7_O9vldVfw accessed 07.11.2014, 2014.

[309] NASA Glenn Research Center, "OTIS - background." Available online http://otis.grc.nasa.gov/background.html accessed 06.02.2014, 2008.

[310] National Aeronautics and Space Administration, "The crew of the Challenger shuttle mission in 1986." Available online http://history.nasa.gov/Biographies/challenger.html accessed 08.12.2015, 2004.

[311] National Aeronautics and Space Administration, "Falcon9launchvehicle NAFCOMcostestimates." Lecture, 2011.

[312] National Business Aviation Association, "NBAA - request for proposals: Aircraft charter." Available online http://www.nbaa.org/admin/options/charter/request-for-proposal.php accessed 10.14.2014, 2013.

[313] Navidi, W., *Statistics for Engineers and Scientists*. McGraw-Hill, 2012.

[314] Negroni, C., "Before flying car can take off, theres a checklist." Available online http://www.nytimes.com/2012/04/29/automobiles/before-flying-car-can-take-off-theres-a-checklist.html?_r=0 accessed 03.03.2016, 2012.

[315] Nelson, D., "Qualitative and quantitative assessment of Optimal Trajectories by Implicit Simulation (OTIS) and Program to Optimize Simulated Trajectories (POST)," Master's thesis, Georgia Institute of Technology, 2001.

[316] Newhouse, J., "A sporty game I-Betting the company," June 1982.

[317] News Discovery, "Flying car gets FAA approval." Available online http://news.discovery.com/autos/flying-car-gets-faa-approval.htm accessed 02.01.2016, 2013.

[318] Niazi, A., Dai, J. S., Balabani, S., and Seneviratne, L., "Product cost estimation: Technique classification and methodology review," *Journal of Manufacturing Science and Engineering*, vol. 128, pp. 563 – 575, 2006.

[319] Nicolai, L. M., *Fundamentals of Aircraft Design*. AIAA Education Series, 2010.

[320] Nieroski, J. S. and Friedland, E. I., "Liquid rocket engine cost estimating relationships," in *AIAA Second Annual Meeting*, 1965.

[321] Nita, M. and Scholz, D., "Estimating the Oswald factor from basic aircraft geometrical parameters," in *Deutscher Luft- und Raumfahrtkongress*, 2012.

[322] Oberle, H. J. and Grimm, W., "BNDSCO: A program for the numerical solution of optimal control problems," tech. rep., Deutsche Forschung Und Versuchsanstalt Fur Luft - Und Raumfahrt (DFVLR), 2001.

[323] Office of Commercial Space Transportation, "Hazard analysis of commercial space transportation," tech. rep., Department Of Transportation, 1995.

[324] Ogden, C. L., Fryar, C. D., Carroll, M. D., and Flegal, K. M., "Mean body weight, height, and body mass index, United States 1960-2002," tech. rep., U.S. Department of Health and Human Services, 2004.

[325] Olds, J., "System sensitivity analysis applied to the conceptual design of a dual-fuel rocket SSTO," in *5th AIAA/NASA/USAF/ISSMO Symposium on Multidisciplinary Analysis and Optimization*, 1994.

[326] Olds, J. R., *Multidisciplinary Design Techniques Applied to Conceptual Aerospace Vehicle Design*. PhD thesis, School of Aerospace Engineering, North Carolina State University, 1993.

[327] Otto, K. and Wood, K., *Product Design*. Prentice Hall, Upper Saddle River, 2001.

[328] PAL-V, "PAL-V One - target specifications." Available online http://pal-v.com/the-pal-v-one/specifications/ accessed 10.14.2014, 2014.

[329] Parkin, M., Powell, M., and Matthews, K., *Economics (6th Edition)*. Addison Wesley, 2005.

[330] Parlos, P. M., *Multi-Criteria Decision Making Methods: A comparing Study*. Kluwer Academic Publishers, 2000.

[331] Passikoff, R., "The new (and superfluous) smartwatch: Everything you ever imagined and nothing you really want." Available online http://www.forbes.com/sites/robertpassikoff/2013/12/18/the-new-and-superfluous-smartwatch-everything-you-ever-imagined-and-nothing-you-really-want accessed 03.03.2016, 2013.

[332] Pelton, J., Logsdon, J., Smith, D., MacDoran, P., and Caughran, P., "Space safety: Vulnerabilities and risk reduction in U.S. space flight programs," tech. rep., Space and Advanced Communications Research Institute (SACRI), 2005.

[333] Pera, R. J. and Jensen, S. C., "Airplane/engine optimization for an operational lift/cruise V/STOL airplane," in *AIAA V/STOL Conference*, 1977.

[334] Perkins, E., "Global survey of passengers gives glimpse of air travel in 10 years," 2015.

[335] Peterson, C., "APEX survey insights: Passenger satisfaction in-flight." Available online `http://apex.aero/2014/10/apex-survey-insights-passenger-satisfaction-in-flight/` accessed 02.02.2015, 2014.

[336] Peterson, J. L., "Petri nets," *ACM Computing Surveys*, vol. 9, no. 3, pp. 223 – 252, 1977.

[337] Petri, C. A., "Fundamentals of a theory of asynchronous information flow," in *IFIP Congress*, 1962.

[338] Philipps, W. F. and Snyder, D. O., "Modern adaptation of Prandtl's classic lifting-line theory," *Journal of Aircraft*, vol. 37, no. 4, pp. 662 – 670, 2000.

[339] Pierson, B. L. and Ong, S. Y., "Minimum-fuel aircraft transition trajectories," *Mathematical and Computer Modelling: An International Journal*, vol. 12, no. 8, pp. 925 – 934, 1989.

[340] Pinon, O. J., *A Methodology for the Evaluation and Selection of Adaptable Technology Portfolios and its Applications to Small and Medium Airports*. PhD thesis, School of Aerospace Engineering, Georgia Institute of Technology, 2012.

[341] Pinon, O. J., Garcia, E., and Mavris, D. N., "A methodological approach for airport technology evaluation and selection," in *26th Congress of International Council of the Aeronautical Sciences (ICAS)*, 2008.

[342] Polsgrove, T., Hopkins, R. C., Thomas, D., Crane, T. M., and Kros, L. D., "Comparison of performance predictions for new low-thrust trajectory tools," tech. rep., Marshal Space Flight Center, 2007.

[343] Ponomarenko, A., "Rocket propulsion analysis." Available online `http://www.propulsion-analysis.com` accessed 05.05.2015, 2014.

[344] Popular Mechanics, "XCOR Lynx: Don't sleep on the space corvette." Available online `http://www.popularmechanics.com/space/a7968/xcor-lynx-dont-sleep-on-the-space-corvette-11644975/` accessed 09.25.2015, 2012.

[345] Powers, B. G., "Space shuttle longitudinal landing flying qualities," *Journal of Guidance*, vol. 9, no. 5, pp. 566 – 572, 1985.

[346] Prasadh, N., Moss, R., Collett, K., Nelessen, A., Edwards, S., and Mavris, D. N., "A systematic method for sme-driven space system architecture down-selection," in *AIAA SPACE 2014 Conference and Exposition*, 2014.

[347] PRICE Systems, LLC, "PRICE® cost models for top down estimating." Available online http://www.pricesystems.com/en-us/offerings/pricecostmodels.aspx accessed 07.11.2014, 2014.

[348] PTC Inc., "PTC - case studies." Available online http://creo.ptc.com/customers-in-action/ accessed 09.29.2015, 2015.

[349] Pugh, S., *Creating Innovative Products Using Total Design*. Addison-Wesley, first ed., 1996.

[350] Pukite, P. and Pukite, J., *Markov Modeling for Reliability Analysis*. Wiley-IEEE Press, 1998.

[351] Quintero, R., *Techno-Economic and Environmental Risk Assessment of Innovative Propulsion Systems for Short-Range Civil Aircraft*. PhD thesis, Cranfield University, 2009.

[352] Rallabhandi, S. K. and Mavris, D. N., "An unstructured wave drag code for the preliminary design of future supersonic aircraft," in *33rd AIAA Fluid Dynamics Conference and Exhibit*, 2003.

[353] Rasiah, D., "A review on the application of investment decision rules applied when considering portfolio selection," *International Journal of Business Strategy*, vol. 12, no. 2, p. 131, 2012.

[354] Raymer, D. P., *Aircraft Design: A Conceptual Approach, Fourth Edition*. AIAA Education Series, 2006.

[355] Reddy, S. Y., Fertig, K. W., and Smith, D., "Constraint management methodology for conceptual design tradeoff studies," in *ASME Design Engineering Technical Conferences and Computers in Engineering Conference*, 1996.

[356] Redhammer Consulting Ltd, "Tornado." Available online http://www.redhammer.se/tornado/ accessed 08.17.2014, 2010.

[357] Rees, D., *COSPAR International Reference Atmosphere: 1986 Part 1: Thermosphere*. Pergamon Press, 1990.

[358] Riehl, J. P., Paris, S., and Sjauw, W., "Comparison of implicit integration methods for solving aerospace trajectory optimization problems," in *AIAA/AAS Astrodynamics Specialist Conference and Exhibit*, 2006.

[359] Robertson, B., Crowley, D., Douglas, R., Mavris, D. N., and Hellman, B., "Configuration determination of a reusable first stage booster utilizing monte carlo based inverse design," in *AIAA SPACE 2011 Conference & Exposition*, 2011.

[360] Rocker, M., "Modeling of nonacoustic combustion instability in simulations of hybrid motor tests," tech. rep., National Aeronautics and Space Administration, 2000.

[361] Rocket and Space Technology, "Rocket propellants." Available online http://www.braeunig.us/space/propel.htm accessed 08.10.2015, 2008.

[362] Rocketplane Global Inc., "Rocketplane XP." Available online http://www.rocketplane.com/technology.html accessed 09.15.2015, 2007.

[363] Rogers, A. W., "Wave drag of arrow wings with modified double-wedge section," tech. rep., Hughes Aircraft Company, 1958.

[364] Rogers, A., *Flying Cars for Everyone in the Near Future*. America Star Books, 2012.

[365] Rogers, C. E., "The solid rocket motor - part 4: Departures from ideal performance for conical nozzles and bell nozzles, straight-cut throats and rounded throats," tech. rep., High Power Rocketry, 2004.

[366] Rohrschneider, R., "Development of mass estimating relationship database for launch vehicle conceptual design," Master's thesis, Georgia Institute of Technology, 2002.

[367] Rongier, I., "Guidelines for the safe regulation, design and operation of suborbital vehicles," tech. rep., International Association for the Advancement of Space Safety, 2013.

[368] Roskam, J., *Airplane Design Part V: Component Weight Estimation*. Darcorporation, 1999.

[369] Roskam, J., *Airplane Design: Preliminary Calculation of Aerodynamic, Thrust and Power Characteristics*. Design, Analysis and Research Corporation, 2000.

[370] Roth, K., *Design catalogues and their usage*. Springer London, 2002.

[371] Roy, R., *Cost Engineering: Why, What and How?* Decision Engineering Report Series, 2003.

[372] Ruijters, E. and Stoelinga, M., "Fault tree analysis: A survey of the state-of-the-art in modeling, analysis and tools," *Computer Science Review*, vol. 15, no. 62, pp. 29 – 62, 2015.

[373] Rutowski, E. S., "Energy approach to the general aircraft performance problem," *Journal of the Aeronautical Sciences (Institute of the Aeronautical Sciences)*, vol. 21, no. 3, pp. 187 – 195, 1954.

[374] S3 - Mission & Goals, "Swiss space systems." Available online `http://www.s-3.ch/en/mission-goals` accessed 09.20.2014, 2003.

[375] Sadraey, M. H., *Aircraft Design: A Systems Engineering Approach*. Wiley - Aerospace Series, 2012.

[376] Sarigul-Klijn, M. and Sarigul-Klijn, N., "A study of air launch method for RLVs," in *AIAA Space 2001 - Conference and Exposition*, 2001.

[377] Sarigul-Klijn, M. and Sarigul-Klijn, N., "Flight mechanics of manned sub-orbital reusable launch vehicles with recommendations for launch and recovery," in *41st Aerospace Sciences Meeting and Exhibit*, 2003.

[378] Sarigul-Klijn, N. and Sarigul-Klijn, M., "High propellant mass fraction hybrid rocket propulsion," 2007.

[379] Scaled Composites, "SpaceShipOne and White Knight." Available online http://www.scaled.com/projects/tierone/ accessed 09.15.2015, 2014.

[380] Scaled Composites, LLC, "Scaled Composites." Available online http://www.scaled.com/hires_gallery/gallery/X-Prize_1/single/XPrize_X1_0178 accessed 04.21.2014, 2014.

[381] Schmitt, R. L., Foreman, K. C., Gertsen, W. M., and Johson, P. H., "Weight estimation handbook for light aircraft," tech. rep., Cessna Aircraft Company, 1959.

[382] Scholz, C., "SpaceShipOne: Re-design drafting drawings." Available online http://curtisscholz.com/?page_id=141 accessed 03.03.2015, 2015.

[383] Schoneman, S. and Buckley, S., "Orbital Suborbital Program (OSP) "Minotaur" space launch vehicle: Low cost space lift for small satellites using surplus minuteman motors," in *AIAA Space 2000 Conference and Exposition*, 2000.

[384] Schrage, D. P. and Mavris, D. N., "Integrated design and manufacturing for the high speed civil transport," in *AIAA Aircraft Design, Systems and Operations Meeting*, 1993.

[385] Schrage, D. P. and Mavris, D. N., "Integrated product/process design/development (IPPD) through robust design simulation," in *1st AIAA Aircraft Engineering, Technology, and Operations Congress*, 1995.

[386] Schrage, D., "Define the problem through Quality Function Deployment (QFD) - IPPD step two." University Lecture, 2013.

[387] Schummer, J., Taylor, N., and MacLennan, B., "Aesthetic values in technology and engineering design," *Philosophy of Technology and Engineering Sciences*, vol. 9, no. 1, pp. 1031 – 1068, 2009.

[388] Schwartz, E. and Kroo, I. M., "Aircraft design: Trading cost and climate impact," in *47th AIAA Aerospace Sciences Meeting Including The New Horizons Forum and Aerospace Exposition*, 2009.

[389] Seebass, A. R., *Space Economics*. American Institute of Aeronautics and Astronautics, 1992.

[390] Seeman, R., "Modeling the life-cycle cost of jet engine maintenance," tech. rep., TU Delft, 2010.

[391] Seibold, R. W., Vedda, J. A., Penn, J. P., BARR, S. E., Kephart, J. F., Law, G. W., Richardson, G. G., Pelton, J. N., Hertzfeld, H. R., Logdson, J. M., and J. A. Hoffman, M. E. L., "Analysis of human space flight safety," tech. rep., The Aerospace Corporation, 2008.

[392] Seshan, B., "Flying car cleared for road use; to hit market by 2012." Available online http://www.ibtimes.com/flying-car-cleared-road-use-hit-market-2012-photos-707705 accessed 02.02.2016, 2011.

[393] Sgobba, T., "Suborbital spaceflight: Do we need accidents?." Available online http://www.spacesafetymagazine.com/space-disasters/virgin-galactic/suborbital-spaceflight-need-accidents/ accessed 03.25.2016, 2014.

[394] Sharpe, W. F., "Capital asset price: A theory of market equilibrium under conditions of risk," *Journal of Finance*, vol. 19, pp. 425 – 442, September 1964.

[395] Shevell, R. S., *Fundamentals of Flight*. PrenticeHall, 1989.

[396] Siemens, "NX aerospace customers." Available online `http://www.plm.automation.siemens.com/en_us/aerospace-defense/space-systems/index.shtml` accessed 09.25.2015, 2015.

[397] Simpson, T. W., Chen, W., Allen, J. K., and Mistree, F., "Conceptual design of a family of products trough the use of the robust concept exploration method," in *AIAA/USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization*, 1996.

[398] Sinclair Vehicles, "Sinclair C5, a new power in personal transport," tech. rep., Sinclair Vehicles, January 1985.

[399] Singiresu, R. S., *The Finite Element Method in Engineering*. Elsevier, fifth ed., 2011.

[400] Smith, J. C., Viken, J. K., Guerreiro, N. M., Dollyhigh, S. M., Fenbert, J. W., Hartman, C. L., Kwa, T.-S., and Moore, M. D., "Projected demand and potential impacts to the national airspace system of autonomous, electric, on-demand small aircraft," in *12th AIAA Aviation Technology, Integration, and Operations (ATIO) Conference and 14th AIAA/ISSM*, 2012.

[401] Smithsonian Air and Space Museum, "Fulton airphibian FA-3-101." Available online `http://airandspace.si.edu/collections/artifact.cfm?object=nasm_A19600127000` accessed 10.14.2014, 2014.

[402] Sniedovich, M., "The art and science of modeling decision-making under severe uncertainty," *Decision Making in Manufacturing and Services*, vol. 1, no. 1, pp. 111 – 136, 2007.

[403] Soban, D. S. and Mavris, D. N., "Assessing the impact of technology on aircraft systems using technology impact forecasting," *Journal of Aircraft*, vol. 50, no. 5, pp. 1380 – 1393, 2013.

[404] Soban, D. S. and Upton, E., "Design of a UAV to optimize use of fuel cell propulsion technology," in *Infotech@Aerospace*, 2005.

[405] Sobek, D. K., Ward, A., and Liker, J., "Toyota's principles of set-based concurrent engineering," *Sloan Management Review*, vol. 40, no. 2, pp. 67 – 83, 1999.

[406] Sommer, S. C. and Short, B. J., "Free-flight measurements of turbulent-boundary-layer skin friction in the presence of severe aerodynamic heating at mach numbers from 2.8 to 7.0," tech. rep., National Advisory Committee for Aeronautics, 1955.

[407] Song, L., "NGPM – a NSGA-II program in Matlab," tech. rep., Aerospace Structural Dynamics Research Laboratory - College of Astronautics, Northwestern Polytechnical University, China, 2011.

[408] Song, L., "NGPM – a NSGA-II program in Matlab v1.4." Available online http://www.mathworks.com/matlabcentral/fileexchange/31166-ngpm-a-nsga-ii-program-in-matlab-v1-4 accessed 04.22.2014, 2011.

[409] Space Future Consulting, "Space vehicles." Available online http://www.spacefuture.com/vehicles/designs.shtml accessed 04.22.2014, 2012.

[410] Space Safety Magazine, "Rocket motor design." Available online http://www.spacesafetymagazine.com/aerospace-engineering/rocketry/hybrid-rocket-overview-part-2/ accessed 08.15.2015, 2014.

[411] SpaceX, "The facts about SpaceX costs." Available online http://www.spacex.com/usa.php accessed 04.09.2014, 2011.

[412] Stamatelatos, M., Vesely, W., Dugan, J., Fragola, J., Minarick, J., and Railsback, J., "Fault tree handbook with aerospace applications," tech. rep., National Aeronautics and Space Administration, 2002.

[413] Stanley, D., Cook, S., and Connolly, J., "NASA's exploration systems architecture study final report," tech. rep., National Aeronautics and Space Administration, 2005.

[414] Stanley, D. O., Talay, T. A., and Lepsch, R. A., "Conceptual design of a fully reusable manned launch system," *Journal of Spacecraft and Rockets*, vol. 29, no. 4, pp. 529 – 537, 1992.

[415] Stanley, D. O., Talay, T. A., Lepsch, R. A., Morris, W. D., and Wurster, K. E., "Conceptual design of a next-generation, fully reusable manned launch system," in *29th AIAA Aerospace Sciences Meeting*, 1991.

[416] Stark Aerospace, Inc., Analytical Methods Division, "MGAERO." Available online http://www.ami.aero/software-computing/amis-computational-fluid-dynamics-tools/mgaero/ accessed 08.16.2014, 2014.

[417] Stark Aerospace, Inc., Analytical Methods Division, "VSAERO." Available online http://www.ami.aero/software-computing/amis-computational-fluid-dynamics-tools/vsaero/ accessed 08.17.2014, 2014.

[418] Starke, J., Belmont, J.-P., Longo, J., Novelli, P., and Kordulla, W., "Some considerations on suborbital flight in Europe," in *15th AIAA International Space Planes and Hypersonic Systems and Technologies Conference*, 2008.

[419] Staton, R., "Statistical weight estimation methods for fighter/attack aircraft," tech. rep., Vought Corporation, 1968.

[420] Staton, R., "Cargo/transport weight estimation methods," tech. rep., Vought Corporation, 1969.

[421] Stivers, L. S. and Spener, B., "Studies of optimum body shapes at hypersonic speeds," tech. rep., National Aeronautics and Space Administration, 1967.

[422] Subchan, S. and Zbikowski, R., "Minimum-time optimal trajectories for the terminal bunt manoeuvre," in *AIAA Guidance, Navigation, and Control Conference and Exhibit*, 2005.

[423] Sutton, G. and Biblarz, O., *Rocket Propulsion Elements*. John Wiley and Sons, Inc., seventh ed., 2001.

[424] Svoboda, C., "Turbofan engine database as a preliminary design tool," *Aircraft Design*, vol. 3, pp. 17 – 31, 2000.

[425] Talay, T., "ME250 launch vehicle design class notes." University Lecture, 1992.

[426] Tan, K. C., Lee, T. H., and Khor, E. F., "Evolutionary algorithms for multi-objective optimization: Performance assessments and comparisons," *Artificial Intelligence Review*, vol. 17, no. 4, pp. 253 – 290, 2002.

[427] Taylor, J., *Jane's All the World Aircraft*. Jane's, 1976.

[428] Taylor, J. W. R., *Jane's All the World's Aircraft 1961-62*. Sampson Low, Marston and Co, 1961.

[429] Taylor, P., *The American Heritage Dictionary of the English Language*. Houghton Mifflin Harcourt, 2011.

[430] Terrafugia, "Terrafugia - The Transition." Available online `http://www.terrafugia.com/aircraft/transition` accessed 10.14.2014, 2013.

[431] Terrafugia, "These are the vehicles that will take you to space." Available online `http://www.terrafugia.com/tf-x` accessed 03.13.2016, 2013.

[432] Tétrault, P.-A., *Numerical Prediction of the Interference Drag of a Streamlined Strut Intersecting a Surface in Transonic Flow*. PhD thesis, Virginia Polytechnic Institute and State University, 2000.

[433] The Boeing Company, "Sparse Optimal Control Software (SOCS)." Available online `http://www.boeing.com/boeing/phantom/socs/capabilities.page?` accessed 07.07.2014, 2014.

[434] The Standish Group, "The chaos report 1994," tech. rep., The Standish Group, 1995.

[435] The Tauri Group, "Suborbital reusable vehicles: A 10-year forecast of market demand," tech. rep., The Tauri Group, 2012.

[436] The University of Edinburgh 2014, "FLITE3D: Software for aerodynamics." Available online `https://www.epcc.ed.ac.uk/projects-portfolio/flite3d-software-aerodynamics` accessed 08.16.2014, 2014.

[437] The Willis Law Firm, "Aviation engines." Available online `http://www.helicoptercrashes.com/aviation-engines` accessed 07.27.2014, 2014.

[438] Thomas, V. K. and Mavris, D. N., "Non-optimal configuration selection during conceptual design," in *11th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference*, 2006.

[439] Thrash, T. A. and McAlister, P. R., "Future markets and economics of suborbital space: Can it reach orbit?," in *Space 2004 Conference and Exhibit*, 2004.

[440] Tieu, B., Kropp, J., and Lozzi, N., "The unmanned space vehicle cost model - past, present, and future," in *AIAA Space 2000 Conference & Exposition*, 2000.

[441] Torenbeek, E., "Synthesis of subsonic airplane design," tech. rep., Delft University, 1996.

[442] Torenbeek, E., *Synthesis of Subsonic Airplane Design*. Delft University Press, 1982.

[443] Trivailo, O., Sippel, M., and Sekercioglu, Y. A., "Review of hardware cost estimation methods, models and tools applied to early phases of space mission planning," *Progress in Aerospace Sciences*, vol. 53, pp. 1 – 17, 2012.

[444] Tumer, I. and Bajwa, A., "Learning about how aircraft engines work and fail," in *35th AIAA/ASME/SAE/ASEE Joint Propulsion Conference and Exhibit*, 1999.

[445] Turchi, P., *Propulsion Techniques, Action and Reaction*. Library of Congress Cataloging-in-Publication Data, 1998.

[446] Ulrich, K. T., *Design: creation of artifacts in society*. Pontifica Press, 2006.

[447] U.S. Department of Transportation - Federal Aviation Administration, "Airworthiness certification." Available online `http://www.faa.gov/aircraft/air_cert/airworthiness_certification/` accessed 10.15.2014, 2014.

[448] U.S. Department of Transportation - Federal Aviation Administration, "FAA regulations." Available online `http://www.faa.gov/regulations_policies/faa_regulations/` accessed 10.15.2014, 2014.

[449] U.S. Department of Transportation - Federal Aviation Administration, "Light-sport aircraft." Available online `http://www.faa.gov/aircraft/gen_av/light_sport/` accessed 10.15.2014, 2014.

[450] van Wijk, J., "The value of visualization," in *IEEE Visualization 2005*, pp. 79 – 86, 2005.

[451] Vanderplaats, G. N., *Multidiscipline Design Optimization*. Vanderplaats Research & Development, Inc., 2007.

[452] Vardaro, M. J., "LeMessurier stands tall: A case study in professional ethics," tech. rep., AIA Trust, 2013.

[453] Vasigh, B., Taleghani, R., and Jenkins, D., *Aircraft Finance, Strategies for Managing Capital Costs in a Turbulent Industry*. J. Ross Publishing, 2012.

[454] Villeneuve, F., *A Method for Concept and Technology Exploration of Aerospace Architectures*. PhD thesis, Georgia Institute of Technology, 2007.

[455] Villeneuve, F. and Mavris, D. N., "A new method of architecture selection for launch vehicles," in *AIAA 13th International Space Planes and Hypersonics Systems and Technologies*, 2005.

[456] Vinciguerra, T., "Flying cars: An idea whose time has never come." Available online `http://www.nytimes.com/2009/04/12/weekinreview/12vinciguerra.html?_r=0` accessed 10.14.2014, 2009.

[457] Virgin Galactic, "SpaceShipTwo: An introductory guide for payload users," tech. rep., Virgin Galactic, 2013.

[458] Virgin Galactic, "These are the vehicles that will take you to space." Available online http://www.ainonline.com/aviation-news/business-aviation/2014-10-14/new-gulfstreams-deliver-more-range-and-cabin-comfort accessed 03.03.2016, 2014.

[459] Virgin Galactic, "Virgin Galactic - booking." Available online http://www.virgingalactic.com/booking/ accessed 05.08.2014, 2014.

[460] Virgin Galactic, "Virgin galactic - our vehicles." Available online http://www.virgingalactic.com/human-spaceflight/our-vehicles/ accessed 01.11.2016, 2014.

[461] Virgin Galactic, "SpaceShipTwo." Available online http://virgingalactic.com/assets/uploads/2014/11/VG_PUG_WEB004_2013061.pdf accessed 09.15.2015, 2015.

[462] Virgin Galactic, "These are the vehicles that will take you to space." Available online http://www.virgingalactic.com/human-spaceflight/our-vehicles/ accessed 03.03.2016, 2016.

[463] Vissepo, V. J., "Legal aspects of reusable launch vehicles," *Journal of Space Law*, vol. 31, pp. 165 – 217, 2005.

[464] Volovoi, V., "Safety by design and flight certification." University Lecture, 2008.

[465] von der Dunk, F. G., "The delimitation of outer space revisited, the role of national space laws in the delimitation issue," tech. rep., Space and Telecommunations Law Program, 1998.

[466] Von Karman, T., "Turbulence and skin friction," *Journal of Aeronautical Sciences*, vol. 1, pp. 1 – 20, 1934.

[467] Wagner Research Company, "Space tourism industry forecast," tech. rep., Space Travel Consultants International, Ltd., 2014.

[468] Walton, M. A. and Hastings, D. E., "Applications of uncertainty analysis to architecture selection of satellite systems," *Journal of Spacecraft and Rockets*, vol. 41, no. 1, pp. 75 – 84, 2004.

[469] Wang, P., Youn, B. D., Xi, Z., and Kloess, A., "Bayesian reliability analysis with evolving, insufficient, and subjective data sets," *Journal of Mechanical Design*, vol. 131, no. 11, pp. 1 – 11, 2009.

[470] Wang, Y. C., McPherson, K., Marsh, T., Gortmaker, S. L., and Brown, M., "Health and economic burden of the projected obesity trends in the USA and the UK," *The Lancet*, vol. 378, no. 9793, pp. 815 – 825, 2011.

[471] Watson, I., *Applying Case-Based Reasoning: Techniques for Enterprise Systems*. Morgan Kaufman Publishers, 1997.

[472] Webb, D. W., Williams, G. S., Tu, A. Q., Seibold, R. W., Baker, C. E., and Young, R. M., "Market demand methodology for U.S. suborbital reusable launch vehicle industry," in *SPACE Conferences and Exposition*, 2014.

[473] Webber, D., "Tourism and developments issues and challenges," tech. rep., NOVA Science Publishers, 2013.

[474] Weikert, S., "ASTOS a major step to efficient trajectory simulation and optimization." Lecture, 2005.

[475] Weissinger, J., "The lift distribution of swept-back wings," tech. rep., National Advisory Committee for Aeronautics, 1947.

[476] Werner, J., "The design flaw that almost wiped out an NYC skyscraper." Available online `http://www.slate.com/blogs/the_eye/2014/04/17/the_citicorp_tower_design_flaw_that_could_have_wiped_out_the_skyscraper.html` accessed 03.03.2016, 2014.

[477] Wiegand, A., *ASTOS User Manual*. ASTOS Solutions GmbH, 2010.

[478] Wiegand, A. and Weikert, S., "Concept and performance simulation with ASTOS." Lecture, 2012.

[479] Williams, S., "Does this company have a clear path to European anti-obesity approval." Available online `http://www.fool.com/investing/general/2013/09/21/does-this-company-have-a-clear-path-to-european-an.aspx` accessed 11.11.2014, 2013.

[480] Wilson, B. J., "APEX: Travelers want better wifi, more internet access." Available online `http://airwaysnews.com/blog/2014/10/22/passengers-want-better-wifi-more-internet-access-says-report/` accessed 03.03.2015, 2015.

[481] Windhorst, R., "Minimum heating re-entry trajectories for advanced hypersonic launch vehicles," Master's thesis, School of Engineering, Santa Clara University, 1996.

[482] Wolter, D., *Common Security in Outer Space and International Law*. United Nations Institute for Disarmament Research, United Nations, 2006.

[483] World Health Organization, "Travel by air: health considerations." Available online `http://whqlibdoc.who.int/publications/2005/9241580364_chap2.pdf` accessed 11.10.2014, 2007.

[484] X Prize Foundation, "Ansari X Prize." Available online `http://space.xprize.org/ansari-x-prize` accessed 04.18.2014, 2011.

[485] Yoon, K. P. and Hwang, C.-L., *Multiple Attribute Decision Making: An Introduction*. Sage University Paper, 1947.

[486] YouGov, "Expense and security checks most common airplane gripes." Available online `https://today.yougov.com/news/2012/08/31/expense-and-security-checks-most-common-airplane-g/` accessed 07.14.2015, 2012.

[487] Young, D. and Olds, J., "Responsive Access Small Cargo Affordable Launch (RASCAL) independent performance evaluation," in *AIAA/CIRA 13th International Space Planes and Hypersonics Systems and Technologies*, 2005.

[488] Young, D. A., *An innovative methodology for allocating reliability and cost in a lunar exploration architecture*. PhD thesis, Georgia Institute of Technology, 2007.

[489] Younossi, O., Arena, M., Moore, R. M., Lorell, M., Mason, J., and Graser, J. C., "Military jet engine acquisition: Technology basics and cost-estimating methodology," tech. rep., RAND Corporation, 2002.

[490] Yu, X., "Vortex lattice methods." University Lecture, 1998.

[491] Zadeh, L. A., "Fuzzy sets," *Information and Control*, vol. 8, no. 3, pp. 338 – 353, 1965.

[492] Zakaria, N., Nasrum, N., Abu, J., and Jusoh, A., "The advantages, potentials and safety of VTOL suborbital space tourism operations," in *5th International Association for the Advancement of Space Safety*, 2011.

[493] Zandbergen, B., "Propulsion system cost data." Available online `http://www.lr.tudelft.nl/en/organisation/departments/space-engineering/space-systems-engineering/expertise-areas/space-propulsion/design-of-elements/cost/` accessed 03.10.2015, 2003.

[494] Zandbergen, B., "Mass data for solid propellant rocket motors." Available online `http://www.lr.tudelft.nl/en/organisation/departments/space-engineering/space-systems-engineering/expertise-areas/space-propulsion/system-design/analyze-candidates/dry-mass-estimation/chemical-systems/srm-mass-data/` accessed 03.03.2015, 2005.

[495] Zhang, X., Huang, G. H., Chan, C., Liu, Z., and Lin, Q., "A fuzzy-robust stochastic multiobjective programming approach for petroleum waste management planning," *Applied Mathematical Modelling*, vol. 34, pp. 2778 – 2788, 2010.

[496] Zhang, X., Huang, G. H., and Nie, X., "Robust stochastic fuzzy possibilistic programming for environmental decision making under uncertainty," *Science of the Total Environment*, vol. 408, pp. 192 – 201, 2009.

[497] Zimmermann, H. J., "Fuzzy programming and linear programming with several objective functions," *Fuzzy Sets and Systems*, vol. 1, pp. 45 – 55, 1978.

[498] Zwicky, F., *Morphological Method of Analysis and Construction*. New York, Wiley-Interscience, 1948.

[499] Zwicky, F., *Discovery, Invention, Research - Through the Morphological Approach*. Toronto: The Macmillan Company, 1969.

[500] Zwicky, F. and Wilson, A., *New Methods of Thought and Procedure: Contributions to the Symposium on Methodologies*. Springer, 1967.

# VITA

Christopher Pierre Frank was born September 15, 1990 in Forbach, France. He graduated with a Bachelor degree in Aerospace from ISAE-Supaéro and was awarded the Jean Walter Zellidja merit-based scholarship from the Académie française. In 2013, he obtained a Master of Science in Aerospace Engineering from both the Georgia Institute of Technology and ISAE-Supaéro. He pursued his doctoral research within the Aerospace Systems Design Laboratory (ASDL) and graduated from the Georgia Institute of Technology with a PhD in Aerospace Engineering in 2016.

Throughout his journey at the ASDL, he has worked as a graduate researcher on multiple topics including more electric aircraft, propulsion, and decision-making processes. He conducted research for Dassault-Aviation, NASA, and Alstom. Moreover, he participated in two different competitions and won the Dassault-Aviation Prize of the Student Aerospace Challenge and the Fly Your Ideas Video Competition organized by Airbus. When not in front of his computer, Christopher is also an active runner and soccer player. He also holds a Private Pilot License and likes exploring new horizons at the controls of an aircraft.